TUGAS AKHIR - KI141502

APLIKASI PENDETEKSI JATUH MENGGUNAKAN MICROSOFT KINECT DAN DECISION TREE

FAHMY THORIQUL HAQ
NRP 5112100037

Dosen Pembimbing
Anny Yuniarti, S.Kom, M.Comp.Sc
Rully Soelaiman, S.Kom, M.Kom

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUGAS AKHIR - KI141502

APLIKASI PENDETEKSI JATUH MENGGUNAKAN MICROSOFT KINECT DAN DECISION TREE

FAHMY THORIQUL HAQ
NRP 5112100037

Dosen Pembimbing
Anny Yuniarti, S.Kom, M.Comp.Sc
Rully Soelaiman, S.Kom, M.Kom

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT - KI141502

FALL DETECTION APPLICATION USING MICROSOFT KINECT AND DECISION TREE

FAHMY THORIQUL HAQ
NRP 5112100037

Dosen Pembimbing
Anny Yuniarti, S.Kom, M.Comp.Sc
Rully Soelaiman, S.Kom, M.Kom

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

LEMBAR PENGESAHAN

APLIKASI PENDETEKSI JATUH MENGGUNAKAN MICROSOFT KINECT DAN DECISION TREE

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi Grafika dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
FAHMY THORIQUL HAQ
NRP : 5112 100 037

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Anny Yuniarti, S.Kom, M.Comp.Sc. .....
NIP: 19810622 200501 2 002 (pembimbing 1)

Rully Soelaiman, S.Kom, M.Kom .....
NIP: 19700213 199402 1 001 (pembimbing 2)

SURABAYA
JUNI 2016

APLIKASI PENDETEKSI JATUH MENGGUNAKAN MICROSOFT KINECT DAN DECISION TREE

Nama Mahasiswa	:	FAHMY THORIQUL HAQ
NRP	:	5112 100 037
Jurusan	:	Teknik Informatika FTIf-ITS
Dosen Pembimbing 1	:	Anny Yuniarti, S.Kom, M.Comp.Sc
Dosen Pembimbing 2	:	Rully Soelaiman, S.Kom, M.Kom

ABSTRAKSI

Jatuh pada lansia adalah masalah serius karena berisiko berujung pada kematian dini jika terjadi cedera atau pendarahan pada otak. Beberapa penelitian juga menyebutkan terjadinya masalah fisik dan psikis berkaitan ketidakmampuan berdiri setelah jatuh yang berujung terbaring di lantai begitu lama.. Seperti lansia yang tinggal sendiri di apartemen atau rumah akan cenderung memiliki dampak yang lebih parah disebabkan kesusahan mendapat pertolongan. Faktanya, menolong lansia sesaat setelah jatuh dapat meningkatkan taraf bertahan hidup dan kemauan untuk kembali beraktivitas secara normal.

Berbagai metode penelitian dilakukan untuk menawarkan pencegahan terhadap jatuh yang dialami lansia yang telah dipasarkan secara komersial. Kebanyakan dari produk tersebut merupakan sistem yang membutuhkan sebuah alat yang harus digunakan/ditempel pada lansia yang membutuhkan aksi dengan menekan tombol pada alat tersebut secara manual ketika terjadi jatuh. Solusi tersebut dirasa kurang optimal karena mengabaikan kemungkinan hilangnya kesadaran sesaat setelah jatuh.

Tugas akhir kali ini membangun sistem pendeteksi jatuh yang mampu mendeteksi jatuh secara otomatis, dengan memanfaatkan sensor pada Kinect yang akan mendeteksi dan menangkap pergerakan sesaat ketika terjadi jatuh. Aplikasi ini diujikan dengan menggunakan aktor untuk melakukan adegan

jatuh dengan beberapa skenario jatuh yang berbeda-beda. Hasil uji coba akurasi menunjukkan bahwa aplikasi telah berhasil mendeteksi jatuh dengan tingkat akurasi 97,7%.
Kata kunci: ***Kinect, Jatuh, Lansia.***

FALL DETECTION APPLICATION USING MICROSOFT KINECT AND DECISION TREE

Student Name	:	FAHMY THORIQUL HAQ
Student ID	:	5112 100 037
Major	:	Informatics Department FTIf-ITS
Advisor 1	:	Anny Yuniarti, S.Kom, M.Comp.Sc
Advisor 2	:	Rully Soelaiman, S.Kom, M.Kom

ABSTRACTION

Fall experienced by the elderly could be so dangerous which can lead to premature death of injury or bleeding in the brain. Several studies mentioned the occurence of physical and or psychological problems related to inability to get up after a fall resulted lying on the floor for long time. All of these risks are increasing with the environment around. Such as the elderly who live alone in an apartment or house will tend to have a more severe impact due to difficulties getting help around. In fact, give the elderly some helps shortly after fall could increase their survival rate and increase the willingness to move back to normal.

Various methods of research undertaken to offer the prevention of falls system which some of them actually have been produced commercially. Most of the product is a system that requires a tool that should be used / affixed to the elderly who need action by pressing the button on the tool manually when it happens to fall. The solution is less optimal because it ignores the possibility of loss of consciousness shortly after the fall.

This final project develop fall detector system application using Kinect as the main sensor to capture the tracked body movement few moments before fall occurred. The application tested using actors to demonstrate realistic fall events with various scenarios. The result in accuration test shows the algorithm succed to detect fall accurately with 97,7% rate.

Keywords: ***Kinect, Fall Detecting, Elders.***

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul:

APLIKASI PENDETEKSI JATUH MENGGUNAKAN MICROSOFT KINECT DAN DECISION TREE

Melalui lembar ini, penulis ingin menyampaikan ucapan terimakasih dan penghormatan yang sebesar-besarnya kepada:

1. Bapak, Ibu, adik, kakak dan keluarga besar yang selalu memberikan dukungan penuh untuk menyelesaikan tugas akhir ini.
2. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang telah banyak menyampaikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
3. Teman-teman angkatan 2012 Jurusan Teknik Informatika ITS yang telah menjadi teman seperjuangan dalam suka dan duka selama 4 tahun penulis menjalani kuliah.
4. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu per satu.

Bagaimanapun juga penulis telah berusaha sebaik-baiknya dalam menyusun tugas akhir ini, namun penulis mohon maaf apabila terdapat kekurangan yang penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2016

FAHMY THORIQUL HAQ

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE SUMBER

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1. Latar Belakang

Jatuh bisa menjadi masalah serius dengan risiko besar jika dialami oleh seorang lansia. Diperkirakan sekitar satu dari tiga lansia yang berumur 65 tahun ke atas mengalami jatuh tiap tahunnya [1]. Sayangnya hal ini sering diremehkan begitu saja. Padahal beberapa dari mereka yang mengalami jatuh, berujung pada terjadinya patah tulang dan trauma otak yang berpotensi mengakibatkan ketidakmampuan bergerak secara mandiri, diikuti trauma untuk beraktivitas, sehingga semakin meningkatkan risiko terulangnya jatuh itu sendiri. Bahkan, yang lebih berbahaya adalah jatuh bisa berujung pada kematian dini jika terjadi cedera atau pendarahan pada otak [1]. Beberapa penelitian juga menyebutkan terjadinya masalah komplikasi fisik dan psikis berkaitan ketidakmampuan berdiri setelah jatuh yang berujung terbaring di lantai dalam waktu lama [2]. Risiko dan dampak juga akan semakin meningkat dengan kondisi lingkungan di sekitar, seperti lansia yang tinggal sendiri di apartemen atau rumah. Mereka yang tinggal sendiri susah untuk mendapat pertolongan pertama, ketika mengalami jatuh. Dibutuhkan sebuah sistem yang dapat mendeteksi kejadian jatuh secara otomatis sehingga dapat mempercepat waktu pertolongan segera untuk lansia yang mengalami jatuh.

Berbagai penelitian telah dilakukan guna menemukan solusi penanganan risiko yang dialami seorang lansia ketika jatuh. Beberapa di antaranya telah melahirkan produk-produk komersial yang akhirnya diproduksi secara umum. Kebanyakan dari produk tersebut merupakan sistem dengan sebuah perangkat *portable* yang

harus digunakan/ditempel pada tubuh. Di samping itu, untuk memastikan telah mendeteksi jatuh, beberapa sistem membutuhkan aksi dari korban jatuh untuk menekan tombol pada alat tersebut secara manual sesaat setelah terjatuh. Bagaimanapun, Perangkat yang bisa digunakan/ditempel pada tubuh seseorang harus terus-menerus tertempel pada tubuh, sehingga membutuhkan pengisian daya secara berkala yang berisiko untuk dilupakan oleh pengguna. Terlebih lagi, perangkat yang membutuhkan aksi pemicu pengguna akan menjadi kurang efektif apabila pengguna kehilangan kesadaran sesaat setelah terjatuh.

Oleh karena itu, dikembangkan suatu sistem yang akan memberikan beberapa solusi dari beberapa kekurangan yang dimiliki produk sistem yang telah dikembangkan sebelumnya. Dengan memanfaatkan Kinect sebagai sensor untuk menerima gambar berupa *frame to frame* yang kemudian akan diproses menggunakan algoritme *decision tree* dengan beberapa ekstraksi fitur untuk mendeteksi kejadian jatuh. Dengan demikian, sistem ini tidak lagi membutuhkan sebuah perangkat *wearable* yang harus dipasang atau ditempel di tubuh dan tidak perlu melakukan pengisian daya pada perangkat tersebut.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini antara lain adalah sebagai berikut:

1. Bagaimana melakukan ekstraksi fitur untuk mendeteksi kejadian jatuh pada lansia?
2. Bagaimana mengimplementasikan *decision tree* deteksi jatuh pada aplikasi sistem pendeteksi jatuh?
3. Bagaimana merancang dan membangun aplikasi sistem pendeteksi jatuh *realtime* otomatis menggunakan Kinect ?

1.3. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, antara lain:

1. Program ini berbasis *desktop* dengan bahasa pemrograman utama C#, *framework* .NET, IDE Visual Studio 2015 Community, basis data MySQL, dan tentunya Kinect SDK untuk mengembangkan sistem yang menggunakan Kinect.
2. Kinect yang digunakan adalah Kinect V2.
3. Fokus pendeteksian jatuh adalah pada lansia yang tinggal sendiri, sehingga kondisi ideal pendeteksian adalah pada satu ruangan terdapat satu Kinect yang terpasang.
4. Aktivitas yang akan diklasifikasikan sebagai jatuh adalah peruahan gerakan dari posisi berdiri menuju posisi yang lebih rendah dari titik tengah ketika berdiri seperti jongkok, berlutut, maupun terbaring.
5. Pengujian pendeteksian jatuh akan dilakukan oleh peraga dengan berbagai skenario jatuh yang divariasikan sebanyak mungkin sesuai batasan.
6. Pemasangan dan penempatan Kinect adalah di sudut atas suatu ruangan yang memiliki ketinggian sekitar 2-3,5 meter.

1.4. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah sebagai berikut:

1. Memahami ekstraksi fitur untuk mendeteksi gerakan seorang lansia ketika terjatuh.
2. Mengetahui implementasi *decision tree* deteksi jatuh pada aplikasi sistem pendeteksi jatuh.
3. Merancang desain dan implementasi aplikasi sistem pendeteksi jatuh *realtime* otomatis menggunakan Kinect

1.5. Manfaat

Manfaat dari hasil pembuatan tugas akhir ini antara adalah memberikan kemudahan pengawasan dan penjagaan lansia yang

tinggal sendiri di suatu ruangan terhadap kemungkinan mengalami jatuh serta meminimalkan waktu pemberian pertolongan untuk lansia dengan peringatan deteksi jatuh.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. **Penyusunan proposal tugas akhir**

   Tahap awal untuk memulai pengerjaan tugas akhir adalah penyusunan proposal tugas akhir. Proposal tugas akhir yang diajukan memiliki gagasan yang sama dengan tugas akhir ini, yaitu aplikasi pendeteksi jatuh menggunakan Microsoft Kinect dan *decision tree***.**

2. **Studi literatur**

   Pada tahap ini dilakukan pencarian sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai risiko dan bahaya jatuh pada lansia, bagaimana pertolongan cepat dapat mengurangi bahaya dari risiko jatuh pada lansia, bagaimana mendeteksi jatuh, ekstraksi fitur yang dibutuhkan untuk mendeteksi kejadian jatuh, penggunaan Kinect dan menghubungkannya dengan komputer, dan pemrograman Kinect dengan bahasa pemrograman C# menggunakan IDE Visual Studio. Selain berhubungan dengan lingkungan pembangunan aplikasi, dilakukan juga studi literature mengenai *decision tree* pada riset-riset terdahulu yang sama-sama menggunakan Kinect untuk mendeteksi jatuh.

3. **Analisis dan Desain Perangkat Lunak**

   Pada tahap ini dilakukan analisis dan desain untuk perangkat lunak sistem pendeteksi jatuh menggunakan Kinect sebagai sensor dan *decision tree* untuk menentukan terjadinya jatuh atau tidak. Kemudian dilakukan perancangan sistem dengan tahap sebagai berikut:

   a. analisis permasalahan;

b. analisis kebutuhan fungsional aplikasi yang akan dibangun;
c. analisis proses yang akan terjadi pada aplikasi;
d. perancangan dan desain aplikasi;
e. perancangan dan desain *decision tree* pendeteksi jatuh;
f. perancangan antarmuka dalam aplikasi;
g. perancangan proses di dalam aplikasi.

4. Implementasi

Implementasi tugas akhir ini merupakan tahap pengembangkan aplikasi berdasar hasil analissi dan desain pada tahap sebelumnya. Beberapa hal yang diperlukan dalam implementasi ini adalah:

a. IDE Visual Studio 2015 Community.
b. Kemampuan Bahasa Pemrograman C#.
c. Microsoft Kinect V2.
d. Kinect for Windows SDK 2.0.
e. Basis data MySQL

5. Pengujian dan evaluasi

Pengujian dilakukan untuk mengukur fungsionalitas aplikasi dengan melakukan uji coba penggunaan secara mandiri. Selain pengujian fungsionalitas, dilakukan juga pengujian tigkat akurasi. Pengujian akurasi ini dilakukan menggunakan metode *confusion matrix* yang akan mengukur tingkat keakuratan algoritma *decision tree* pendeteksi jatuh pada aplikasi yang telah diimplementasi.

6. Penyusunan buku Tugas Akhir

Pada tahap ini dilakukan proses penyusunan buku yang memuat keseluruhan dokumentasi dan pembuatan aplikasi semua proses yang telah dilakukan beserta hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.7. Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab yang berisi mengenai latar belakang, rumusan permasalahan, batasan masalah, tujuan, manfaat, hingga metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

Bab II Dasar Teori

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini berisi tentang desain sistem, desain basis data, dan desain antarmuka yang akan dibuat.

Bab IV Implementasi

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode sumber yang digunakan untuk proses implementasi.

Bab V Uji Coba Dan Evaluasi

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

Bab VI Kesimpulan Dan Saran

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan tugas akhir ini. Teori-teori tersebut meliputi:

2.1. Jatuh Pada Lansia

Jatuh adalah tiba-tiba, tidak disengaja yang menyebabkan perubahan posisi seseorang berada di area yang lebih rendah, pada suatu objek, di lantai atau di rumput atau di tanah, selain akibat dari serangan paralisis, epilepsi atau kekuatan di luar batas [3]. Dengan kata lain, jatuh dapat diartikan sebagai kejadian tiba-tiba dan tidak disengaja yang mengakibatkan seseorang berada pada posisi yang lebih rendah, baik dalam posisi duduk maupun terbaring di dataran [4]. Pada lansia yang mengalami jatuh, banyak diantaranya mengalami cedera parah, seperti patah tulang pinggul dan trauma, yang berikutnya membuat ketidakmampuan untuk mandiri dan beraktivitas, kemudian akan berujung pada kematian dini [1].

1. Pengertian

   Jatuh pada lansia merupakan jatuh yang dialami oleh korban dengan usia yang telah lanjut. Sehingga, harus diperhatikan mengingat lansia memiliki kerentanan fisik yang lebih tinggi terhadap risiko jatuh.

2. Penyebab Jatuh Pada Lansia

   Penyebab jatuh pada lansia biasanya merupakan gabungan dari beberapa faktor penyebab jatuh tersebut. Antara lain:

   a. Kecelakaan
   b. Nyeri kepala, pusing dan atau vertigo
   c. Hipotensi
   d. Efek obat-obatan
   e. Kehilangan kesadaran secara tiba-tiba

   Beberapa penyebab jatuh di atas merupakan penyebab yang tergabung dari beberapa faktor penyebab jatuh baik

intrinsik seperti kondisi dari dalam diri, maupun ekstrinsik yang berupa faktor dari luar seperti obat-obatan dan lingkungan [4].

3. Dampak Jatuh Pada Lansia

   Sekitar sepertiga populasi lansia yang berumur di atas 65 mengalami jatuh tiap tahun dan frekuensi untuk jatuh semakin meningkat bersama bertambahnya umur. Pada kalangan lansia dengan usia 80 tahun keatas, sekitar setengah dari seluruhnya mengalami jatuh. [5]. Beberapa dampak akibat jatuh pada lansia yaitu:

   a. Semakin sering jatuh.

      Lansia yang pernah mengalami jatuh akan berisiko 2-3 kali untuk jatuh lagi. Karena, kebanyakan kejadian jatuh akan membuat trauma dan takut untuk jatuh. Ketakutan tersebut dapat berujung pada orang itu untuk mengurangi aktivitas sehari-hari mereka. Padahal, tidak banyak bergerak akan menyebabkan otot melemah dan mengecil (atropi), sehingga meningkatkan kemungkinan jatuh lagi [1].

   b. Patah tulang.

      Satu dari lima kejadian jatuh pada lansia menyebabkan cedera serius seperti patah tulang. Sekitar 95% dari patah tulang pinggul pada lansia dikarenakan mengalami jatuh, biasanya jatuh dengan posisi menyamping [1].

   c. Cedera dan pendarahan pada otak.

      Jatuh juga merupakan penyebab cedera pada otak atau *traumatic brain injuries* (TBI) yang paling sering terjadi. TBI pada tingkat terentu bias berujung pada kelumpuhan, apalagi kematian [1].

2.2. Pencegahan dan Pertolongan

Dengan mengetahui fakta-fakta statistik yang telah disebutkan sebelumnya, penting bagi kerabat dan keluarga seorang lansia untuk ikut membantu melakukan baik pencegahan jatuh maupun pertolongan saat mengalami jatuh. Hal seperti demikian

menjadi sangat penting karena beberapa alasan berikut:

1. Jatuh mungkin sering tidak mengakibatkan cedera yang serius pada lansia, namun sekitar 47% lansia yang mengalami jatuh tanpa cedera tidak bisa bangkit berdiri lagi tanpa bantuan setelah terjatuh [1].
2. Ketika seorang lansia mengalami jatuh dan tidak dapat bangkit berdiri sendiri, dalam jangka waktu sesaat akan mengalami ketidakmampuan bergerak dan tergeletak begitu saja di lantai. Hal tersebut ternyata dapat memengaruhi kesehatan fisik. Sel otot akan terpecah pada 30-60 menit setelah jatuh yang akan mengakibatkan dehidrasi, hipotermia, sampai pneumonia [5].

Oleh karena itu, penting peranan orang di sekitar untuk memberikan bantuan pada lansia, baik berupa pencegahan terjadinya jatuh maupun pertolongan sesaat setelah terjatuh. Beberapa langkah pencegahan jatuh pada lansia bisa dilakukan dengan beberapa hal. Diantaranya adalah:

1. Mengidentifikasi faktor risiko, penilaian keseimbangan dan gaya berjalan. Bisa dilakukan dengan mengkaji apakah lansia berisiko jatuh atau tidak menggunakan pengkajian skala jatuh Morse (*Morse Fall Scale*) [4] yang ditunjukkan pada Tabel 2.1 dan Tabel 2.2 berikut ini:

Tabel 2.1 Pengkajian skala jatuh Morse (a)

NO	Pengkajian	Skala		Nilai
1	Riwayat jatuh: apakah lansia dalam 3 bulan terakhir?	Tidak	0	
		Ya	25	
2	Diagnosa sekunder: apakah lansia memiliki lebih dari satu penyakit?	Tidak	0	
		Ya	15	
3	Alat Bantu jalan:			
	- *Bed rest*/ dibantu perawat	0		
	- Kruk/ tongkat/ *walker*	15		
	- Berpegangan pada benda-benda di sekitar (kursi, lemari, meja)	30		

Tabel 2.2 Pengkajian skala jatuh Morse (b)

No	Pengkajian	Skala		Nilai
4	Terapi Intravena: apakah saat ini lansia terpasang infus?	Tidak	0	
		Ya	20	
5	Gaya berjalan/ cara berpindah:			
	- Normal/ *bed rest*/ *immobile* (tidak dapat bergerak sendiri)	0		
	- Lemah (tidak bertenaga)	10		
	- Gangguan/ tidak normal (pincang/ diseret)	20		
6	Status Mental:			
	- Lansia menyadari kondisi dirinya	0		
	- Lansia mengalami keterbatasan daya ingat	15		
Total Nilai				
Keterangan Nilai	0-24 = tidak beresiko jatuh 25-50 = risiko rendah ≥ 51 = risiko tinggi untuk jatuh			

2. Memberikan latihan fleksibilitas gerakan, latihan keseimbangan fisik dan koordinasi keseimbangan [4].
3. Mengevaluasi bagaimana keseimbangan badan lansia dalam melakukan gerakan berpindah tempat [4].
4. Selain meningkatkan aspek kebutuhan fisik, penting juga meningkatkan kebutuhan psikologis dan sosial. Dengan anggota keluarga lansia ataupun petugas panti untuk mengunjungi secara rutin untuk mengamati perkembangannya [4].
5. Memperbaiki kondisi lingkungan sekitar lansia yang dianggap tidak aman [4].

Selain pencegahan, pertolongan pada lansia juga perlu dilakukan sesegera mungkin pada lansia yang mengalami jatuh. Memberikan pertolongan pada lansia yang tidak bisa berdiri sendiri setelah terjatuh dapat meningkatkan kelangsungan hidup sekitar

80% dan kemauan untuk kembali beraktivitas mandiri seperti biasa pada lansia tersebut [5].

Langkah-langkah pertolongan yang bisa dilakukan ketika mengetahui lansia yang terjatuh adalah [6]:

1. Apabila lansia bisa bangun kembali, posisikan ke untuk duduk di kursi dengan posisi yang nyaman.
2. Sebaliknya, jika lansia tidak bisa bangun, segera beri pertolongan pertama dan posisikan ke posisi yang nyaman. Jika memungkinkan, tanpa paksaan membantu untuk bangkit lagi ke posisi awal.

2.3. Sistem Pendeteksi Jatuh

Kebanyakan pertolongan dan pencegahan jatuh pada lansia dirasa kurang efektif. Hal ini terjadi karena pemantauan terhadap lansia tidak dapat dilakukan terus-menerus setiap saat. Diperlukan adanya suatu sistem yang dapat mendeteksi gerakan jatuh secara otomatis. Dengan demikian, pemantauan secara konvensional yang merngharuskan seorang pengasuh untuk berada terus disekitar lansia bisa digantikan dengan pemantauan yang *realtime* dan tidak mengharuskan pengawasan dilakukan secara langsung.

Sudah banyak produk yang telah diproduksi secara komersial berupa sistem perawatan lansia yang hadir sebagai inovasi solusi untuk masalah tersebut. Hal ini menunjukkan betapa pentingnya perawatan dan pengawasan dilakukan guna meminimalkan risiko jatuh pada lansia. Sistem-sistem pengawasan lansia tersebut bisa dilihat pada Tabel 2.3 dan Tabel 2.4 di bawah:

Tabel 2.3 Spesifikasi produk komersial sistem perawatan lansia (a)

Merk Varian	**Sense4 care**	**alert1**	**Vigi' Fall**	**Senior Safety**	**AI Home System**
Wearable device	Ya	Ya	Ya	Ya	Tidak
Butuh aksi manual dari pengguna	Tidak	Ya	Tidak harus	Tidak harus	Tidak

Tabel 2.4 Spesifikasi produk komersial sistem perawatan lansia (b)

Merk Varian	**Sense4 care**	**alert1**	**Vigi' Fall**	**Senior Safety**	**AI Home System**
Jumlah perangkat yang dibutuhkan	1 + PC/ ponsel sebagai penerima	2	3	2	2
Baterai	Ya (tahan 3 bulan)	Ya (tahan 2 tahun)	Ya (tahan 3 bulan)	Ya (tahan 5 bulan)	No (AC *Adapter*)
Jangkauan maksimal	Tidak terbatas (dengan GPS)	120 m	60-80 m	300 m	Terbatas pada jangkauan kamera
Ketahanan air	Tidak	Ya	Ya	Ya	Tidak
Website	http://www.sense4care.com/en	https://www.alert-1.com/content/fall-detection-technology	http://www.vigilio.fr/	http://www.seniorsafety.com/medical-alert-system	http://aihomesystems.com/products/helper.html

Jika dilihat, sistem-sistem tersebut menawarkan berbagai kelebihan seperti ketahanan air, daya tahan baterai pada perangkat *wearable* hingga tidak adanya batas jarak maksimum dengan menggunakan GPS. Namun Pada umumnya, sistem yang ditawarkan memerlukan sebuah perangkat *wearable* yang harus digunakan/ ditempelkan pada bagian tubuh lansia. Selain itu, beberapa membutuhkan aksi menekan tombol pada perangkat *wearable* untuk memastikan telah terjatuh. Hal ini tentunya

memunculkan masalah baru lagi karena bagaimanapun perangkat *wearable* membutuhkan untuk diisi ulang secara kontinu agar tetap bisa digunakan. Selain itu, kebutuhan aksi pemicu pada perangkat tersebut juga mengabaikan kondisi tertentu yang tidak terduga seperti apabila lansia baik sesaat sebelum, ketika, maupun setelah jatuh mengalami kehilangan kesadaran, cedera serius, sampai kelumpuhan sesaat.

2.4. Kinect

Kinect merupakan alat kontrol pertama yang dapat mendeteksi gerakan partisipan untuk melakukan sebuah kendali tertentu pada Xbox 360 maupun pada komputer berbasis windows. Kinect telah diluncurkan dalam beberapa versi. Versi terbaru dapat mendeteksi gerakan 6 orang sekaligus, jauh lebih baik daripada versi pendahulunya yang hanya dapat mendeteksi gerakan 2 orang saja. Dengan adanya kamera dan cahaya yang dipancarkan dari Kinect, partisipan hanya perlu menempatkan dirinya di depan Kinect dan bergerak sesuai kehendaknya. Maka kontrol akan terjadi pada program sesuai dengan gerakan yang dilakukan partisipan. Gambar 2.1merupakan contoh *Kinect for Windows*.

Gambar 2.1 Kinect for Windows v2

2.4.1. Jenis dan Versi Kinect

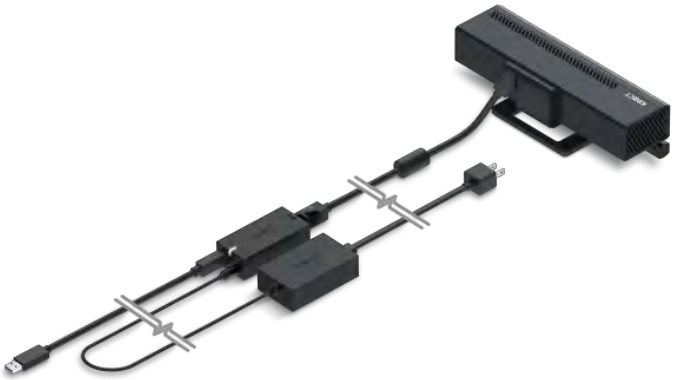
Sensor pada Kinect dapat dianggap sebagai kamera 3D, yang menangkap warna piksel dengan data dari kedalaman setiap piksel. Ada dua jenis produk Microsoft Kinect, yaitu Kinect untuk Xbox 360 dan Kinect untuk Windows. Kedua jenis Kinect tersebut dapat digunakan untuk pengembangan perangkat lunak. Kedua Kinect ini memiliki kamera RGB, sensor kedalaman (IR) dan mikrofon. Kini Kinect telah mencapai versi 2, yang cukup banyak mengalami penambahan kualitas dari versi yang pertama. Namun selanjutnya kita hanya akan membahas Kinect versi 1 karena aplikasi ini menggunakan Kinect versi tersebut. Perbandingan spesifikasi dari kedua versi Kinect for Windows seperti dapat dilihat pada Tabel 2.5Tabel 2.5 berikut [7]:

Tabel 2.5 Perbandingan spesifikasi Kinect For Windows

Feature	Kinect for Windows 1	Kinect for Windows 2
Color Camera	640 x 480 (pixel) @30 fps	1920 x 1080 (pixel) @30 fps
Depth Camera	320 x 240 (pixel)	512 x 424 (pixel)
Max Depth Distance	0.8~3.5m	0.5~8 M
Field of View of Color Image	62°x48.6°	84.1°x53.8°
Field of View of IR Image and Depth View	57.5°x43.5°	70.6°x60°
Method of Depth Measurement	Light coding	Time of Flight
Skeleton Joints Defined	20 joints	25 joints
Full Skeletons Tracked	2	6
USB Standard	2.0	3.0

2.4.2. Spesifikasi Kinect

Sudut yang dapat dijangkau oleh sensor pada Kinect adalah 43^o vertikal dan 57^o horizontal. "Leher" Kinect dapat mengubah jangkauan kurang lebih sebesar 27^o vertikal. Jarak yang dapat dijangkau oleh sensor pada Kinect adalah kisaran 0,8 – 4 meter. *Frame rate* untuk sensor kedalaman dan kamera RGB adalah sebesar 30 *fps*. Masih banyak lagi spesifikasi Kinect yang luar biasa, namun untuk aplikasi ini hanya digunakan kedua sensor yang telah disebutkan di atas, yakni sensor kedalaman dan kamera RGB. Bagian dan komponen dari Kinect yang dibutuhkan dapat dilihat pada Gambar 2.2.



Gambar 2.2 Bagian-bagian perangkat Kinect

2.4.3. Kinect SDK

Microsoft Kinect SDK merupakan pustaka yang bekerja bersama berbagai platform Windows (*WPF* maupun *WinForm*) untuk menghasilkan program aplikasi yang menggunakan Kinect sebagai perangkat masukannya. Saat ini sudah dirilis SDK terbaru versi 2 (untuk Windows 8) dan versi 1.8 (untuk Windows 7). Pada pengembangan sistem kali ini akan menggunakan SDK for Kinect 2.0, karena pengembangan menggunakan perangkat Kinect for Windows 2.0.

2.5. Metode Decision Tree dan Algoritma Pendeteksi Jatuh

Dalam mendeteksi jatuh, diperlukan adanya metodologi yang dapat mendeteksi jatuh dengan tingkat kebenaran yang akurat. Salah satu metode yang dapat diterapkan adalah *decision tree*. Pada dasarnya, *decision tree* merupakan salah satu metode klasifikasi yang populer karena mudah untuk diimplementasi dan dimengerti oleh manusia. *Decision tree* merupakan model prediksi menggunakan struktur pohon atau bisa disebut struktur hirarki. Konsepnya adalah dengan mengubah data-data menjadi pohon keputusan dan aturan-aturan keputusan. Kelebihan lainnya adalah dapat memecah proses pengambilan keputusan yang kompleks menjadi lebih sederhana dengan dibentuknya pohon keputusan tersebut. Dalam membuat *decision tree*, perlu ditentukan atribut-atribut dengan *information gain* yang paling tinggi yang dapat memengaruhi klasifikasi data.

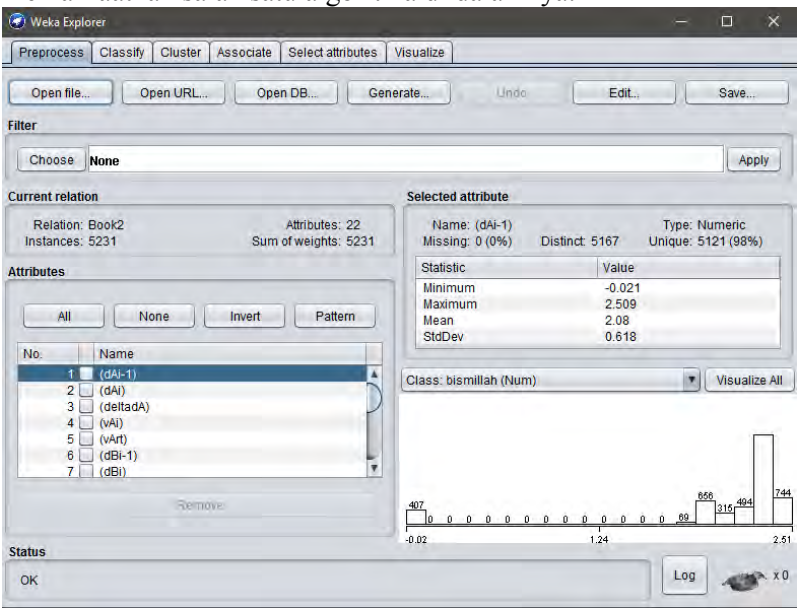
Berdasarkan riset - riset terdahulu yang telah dilakukan, terdapat beberapa algoritma untuk mendeteksi jatuh termasuk didalamnya dengan *decision tree*. Beberapa algoritma tersebut antara lain adalah mendeteksi jatuh dengan dua langkah *processing*. Pada langkah pertama, dilakukan karakterisasi pada posisi *vertical* seseorang yang berdiri untuk tiap *frame* yang kemudian diikuti identifikasi tiap waktu. Setelah itu, pada langkah kedua dilakukan pengekstraksian fitur yang dibutuhkan dan memasukkan ke dalam *decision tree* untuk menentukan terjadinya jatuh [8].

Pada studi literatur lain, algoritma mendeteksi jatuh lebih ditekankan pemanfaatan Kinect sebagai sensor penangkap dilengkapi dengan SDK yang handal. Dengan kemampuan Kinect untuk mendeteksi tubuh manusia, dapat ditentukan titik-titik stres pada tubuh dengan *information gain* tertinggi dalam penentuan kejadian jatuh. Kemudian mencari jarak titik-titik tersebut pada bidang normal dan kecepatan yang berubah tiap waktu untuk menemukan *threshold*. Akhirnya apabila nilai-nilai tersebut melebihi *threshold* bisa ditentukan telah terdeteksi jatuh [9].

2.6. Weka

Waikato Environment for Knowledge Analysis atau lebih biasa disebut Weka adalah sebuah perangkat lunak *open source* untuk keperluan *machine learning* dalam *data mining*. dikembangkan menggunakan bahasa pemrograman Java, Aplikasi ini dikembangkan pertama kali oleh Universitas Waikato yang terletak di Selandia Baru. Weka memiliki banyak algoritma *machine learning* yang bervariasi didalamnya. Selain itu, Weka juga memiliki banyak *tools* tambahan yang berguna untuk pengolahan data mulai dari *pre-processing*, *classification*, *regression*, *clustering*, *association rules*, sampai *visualization*.

Dengan menggunakan Weka, dapat dilakukan sebuah generalisasi dan atau formulasi dari sekumpulan data *sampling* memanfaatkan salah satu algoritma di dalamnya.



Gambar 2.3 Fitur Explorer Weka

Tampilan utama dari aplikasi ini adalah fitur *explorer* seperti ditunjukkan pada Gambar 2.3.

Dalam penggunaannya, semisal terdapat sekumpulan data, dan hendak dibangun sebuah *decision tree* dari sekumpulan data tersebut. Pertama, data-data tersebut perlu disiapkan dengan fitur *explorer* dalam Weka. Kemudian, menentukan metode *decision tree* yang digunakan, eksekusi pembangunan *tree*, dan mengevaluasi hasil keluaran *tree*. Untuk tampilan fitur *explorer* pada Weka dapat dilihat. Pada fiture *explorer* ini, dapat dilakukan pengecekan terhadap hasil keluaran dengan memvisualisasikan atau membandingkan terhadap beberapa sub *dataset*, termasuk dapat dievaluasi kesalahan pengklasifikasian dan model *tree* yang dihasilkan [10].

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis dan perancangan sistem yang akan dibangun. Analisis membahas semua persiapan yang akan menjadi pokok pikiran pembuatan aplikasi ini. Mulai dari masalah yang melatarbelakangi, hingga analisis gambaran awal sistem yang akan dibuat. Perancangan sistem membahas hal-hal yang berkaitan dengan pondasi atau dasar pembuatan aplikasi, yang meliputi perancangan basis data, tampilan antar muka halaman aplikasi, hingga perancangan alur proses yang akan diimplementasikan di dalam aplikasi.

3.1. Analisis

Tahap analisis meliputi analisis masalah, analisis kebutuhan, deskripsi umum sistem, dan kasus penggunaan sistem yang dibuat.

3.1.1. Analisis Permasalahan

Pertolongan pertama untuk lansia yang mengalami jatuh merupakan salah satu langkah terbaik untuk memberikan langkah pencegahan terhadap jatuh-jatuh yang mungkin terjadi berikutnya. Bagaimana tidak, dampak yang disebabkan jatuh kemudian terbaring di lantai dalam waktu yang lama akan memengaruhi kondisi fisik dan atau psikis seperti lumpuh, cedera otak, sampai dengan trauma untuk beraktivitas normal. Biasanya, seorang lansia yang terjatuh dan tidak dapat bangkit bangun lagi akan kesulitan untuk meminta bantuan.

Pada bab sebelumnya, telah disebutkan bahwa memberikan bantuan padan lansia jatuh untuk bangkit kembali akan memberikan dampak sekitar 80% secara psikis maupun fisik pada lansia tersebut untuk tetap bertahan hidup dengan beraktivitas secara mandiri. Hal itu tentunya menjadi sangat penting bagi keluarga atau perawat untuk memberikan pengawasan dan perawatan dengan segera memberikan pertolongan ketika

mengetahui lansia terjatuh, mengingat banyak lansia yang tinggal sendiri dan jauh dari pengawasan sekitar.

Pada kenyataannya, pertolongan hanya akan diberikan ketika seorang perawat ataupun orang di sekitar melihat dan mengetahui kejadian jatuh pada lansia tersebut. Bagaimanapun pengawasan tidak bisa dilakukan terus-menerus karena seorang perawat tidak akan selalu ada di sekitar lansia. Sehingga permasalahan yang dihadapi di sini adalah bagaimana pengawasan bisa dilakukan secara *real-time* dan otomatis tanpa perlu melihat secara langsung untuk mengetahui bahwa lansia mengalami jatuh.

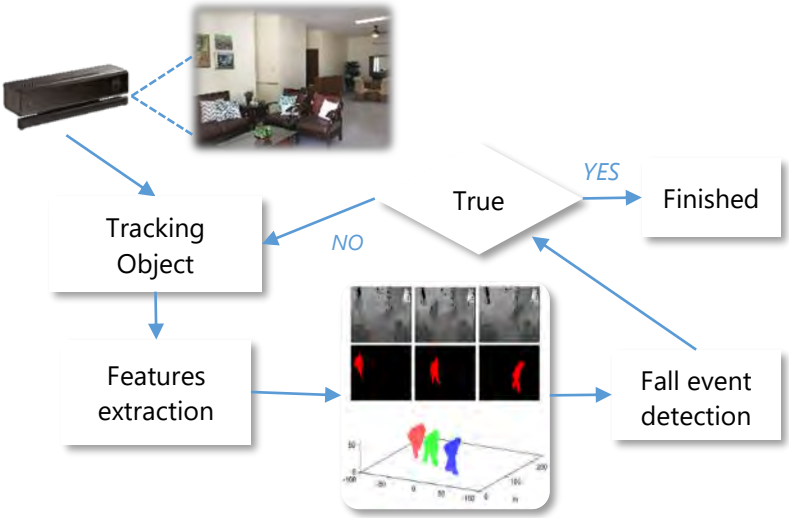
Oleh karena itu, aplikasi pendeteksi jatuh pada lansia ini akan memberikan inovasi solusi untuk memberi pengawasan pada lansia yang hidup di tempat tinggal seorang diri, khususnya apartemen. Memanfaatkan Kinect sebagai sensor utama, sistem yang akan dibangun kali tidak memerlukan suatu perangkat tambahan yang harus dipakai/ ditempel pada tubuh lansia seperti kebanyakan produk-produk perawatan lansia yang telah dipasarkan. Selain itu, beberapa perangkat *portable* bawaan pada sistem tertentu memerlukan aksi pemicu dari lansia pengguna seperti menekan tombol, sehingga secara tidak langsung sistem tersebut mengabaikan kemungkinan hilangnya kesadaran pada lansia sesaat ketika terjatuh. Dengan tidak adanya perangkat *portable* pada aplikasi yang dikembangkan kali ini akan memberi kebebasan pada penggunaan dan implementasinya, mengingat perangkat *portable juga* memerlukan pengisian daya berulang-ulang untuk tetap aktif dalam jangka waktu tertentu.

3.1.2. Deskripsi Umum Sistem

Sistem yang akan dibangun kali ini merupakan aplikasi berbasis *desktop* yang memanfaatkan Kinect sebagai sensor utama. Pengembangan akan dibantu dengan penggunaan Kinect for Windows SDK dengan bahasa C# sebagai bahasa utama.

Aplikasi ditujukan kepada kerabat ataupun perawat yang memberikan pengawasan pada lansia yang tinggal sendiri khususnya apartemen. Lansia yang tinggal sendiri sangat membutuhkan pengawasan setiap saat yang bisa memantau segala

kejadian pada lansia. Hal ini dimaksudkan untuk berjaga-jaga apabila secara tiba-tiba lansia mengalami jatuh dikarenakan penyebab yang variatif. Dengan adanya sistem pendeteksi jatuh yang otomatis, pertolongan akan dapat diberikan dengan sesegera mungkin sesaat setelah kejadian jatuh terdeteksi. Terlebih mengingat adanya kemungkinan besar lansia akan mengalami cedera dan atau ketidakmampuan berdiri bangkit lagi dari lantai.

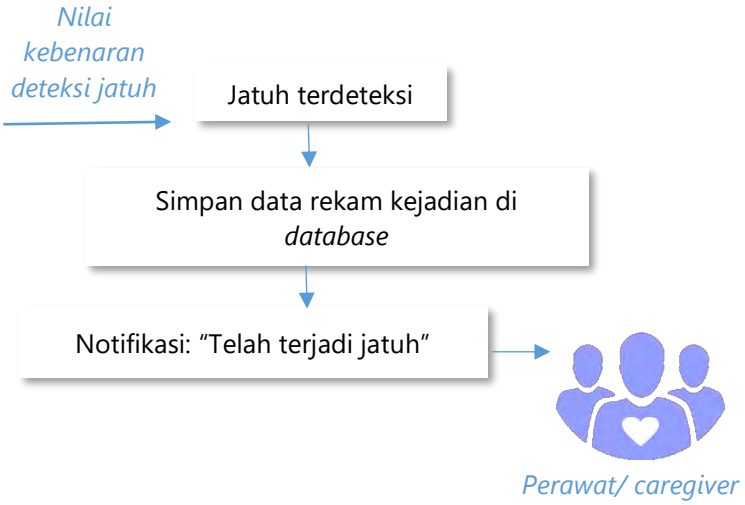


Gambar 3.1 Diagram Alur Fase Pendeteksian Jatuh

Cara kerja sistem aplikasi secara garis besar terbagi menjadi dua fase utama, yaitu:

1. Mendeteksi jatuh, pada fase ini dengan menggunakan teknologi Kinect sebagai sensor pendeteksi kerangka manusia, aplikasi kemudian memproses tangkapan Kinect dengan algoritma *decision tree* yang akan mengekstrak beberapa fitur sehingga menghasilkan *confidence level* yang akurat dalam mendeteksi segala jenis gerakan jatuh. Sesuai Gambar 3.1, keluaran dari fase pertama adalah keputusan ya atau tidak dari terdeteksinya kejadian jatuh

2. Setelah mendeteksi jatuh, pada fase berikutnya aplikasi akan menyimpan rekam histori jatuh dan memberikan alarm peringatan sederhana yang menandakan telah terjadinya jatuh di ruangan di mana Kinect terpasang. Diagram alur fase ini bisa di lihat pada Gambar 3.2



Gambar 3.2 Diagram Alur Fase Notifikasi

3.1.3. Analisis Kebutuhan

Kebutuhan sistem yang akan dibuat pada tugas akhir ini dibagi menjadi dua macam, yaitu kebutuhan fungsional dan kebutuhan non fungsional.

3.1.3.1. Kebutuhan Fungsional

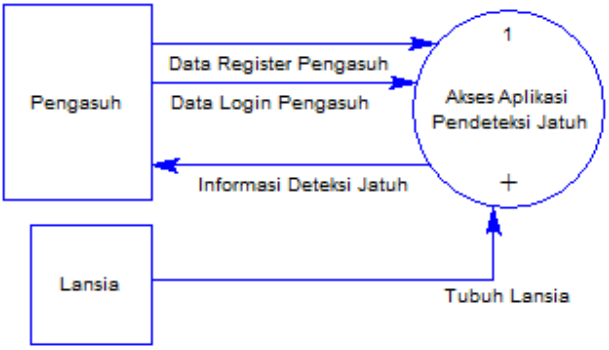
Kebutuhan utama dalam aplikasi ini adalah kemampuan aplikasi dalam mendeteksi terjadinya gerakan jatuh dengan akurasi yang tinggi. Sensor utama untuk menangkap gerakan tersebut akan menggunakan Kinect disertai algoritma *decision tree*. Selain itu, aplikasi juga bisa menyimpan rekam histori jatuh yang menyimpan data kejadian jatuh. Beberapa kebutuhan fungsional dari aplikasi yang akan dibangun pada tugas akhir ini dapat dilihat pada Tabel 3.1 di bawah berikut.

Tabel 3.1 Deskripsi kebutuhan fungsional perangkat lunak

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-001	Melihat penjelasan, deskripsi singkat aplikasi	Pengasuh dapat melihat penjelasan singkat, deskripsi, dan beberapa hal yang berkaitan dengan penggunaan aplikasi.
F-002	Menjalankan mode pengawasan	Pengasuh bisa mengaktifkan mode pengawasan, dimana system mulai merekan pendeteksian jatuh.
F-003	Melihat rekam histori jatuh	Pengasuh bisa melihat record dari pendeteksian jatuh selama sistem dalam mode pengawasan.

3.1.3.2. Aliran Informasi

Aliran informasi untuk aplikasi sistem pendeteksi jatuh dapat dilihat pada Gambar 3.3 berupa *data flow diagram* level 0. Dalam diagram tersebut dapat dilihat pengasuh sebagai pengguna dapat mengakses aplikasi dengan memasukkan data registerasi yang dibutuhkan terlebih dahulu.

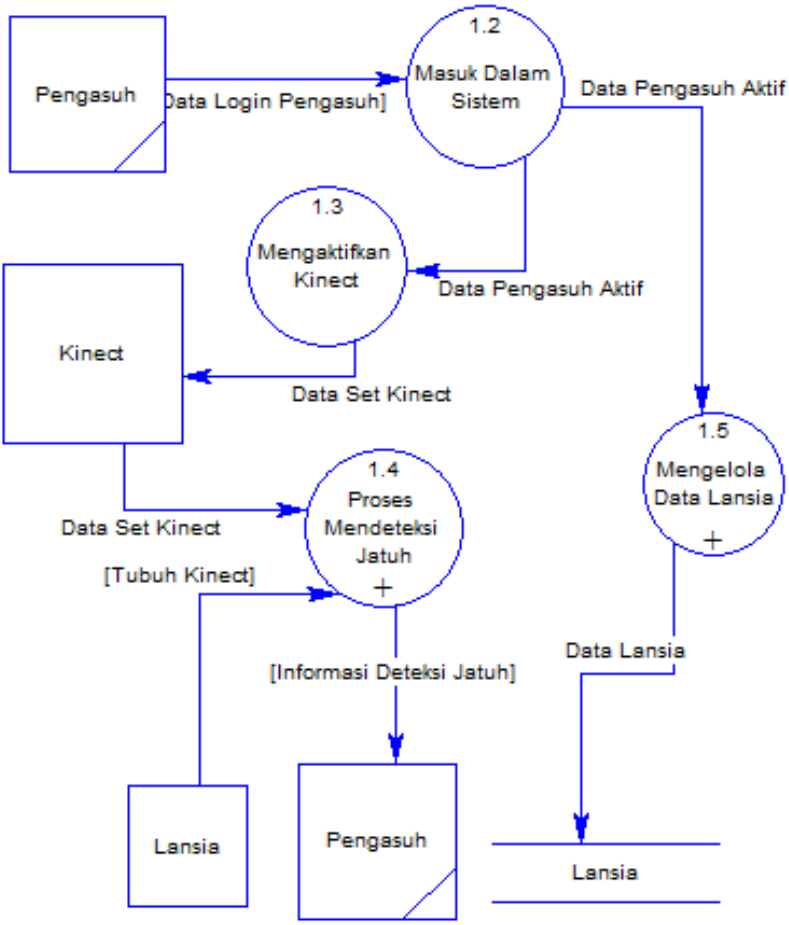


Gambar 3.3 DFD Level 0

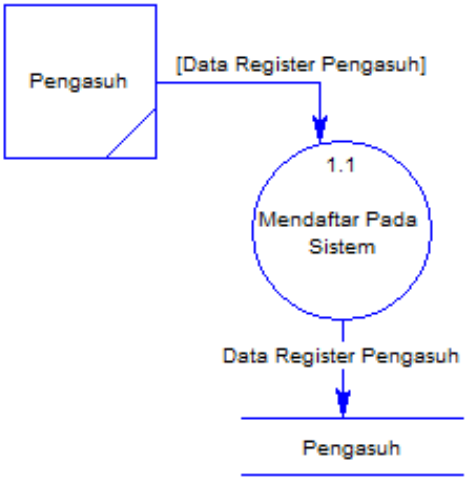
3.1.3.3. DFD Level 1

Dalam DFD Level 1, secara keseluruhan terbagi menjadi dua bagian besar, yaitu ketika pengasuh masuk atau *login* dan ketika pengasuh melakukan registrasi. Setelah melakukan proses

registrasi, pengasuh bisa masuk dan mengakses aplikasi untuk melakukan pengaktifan Kinect dan pengelolaan data lansia. DFD Level 1 ini dapat dilihat pada Gambar 3.4 dan Gambar 3.5



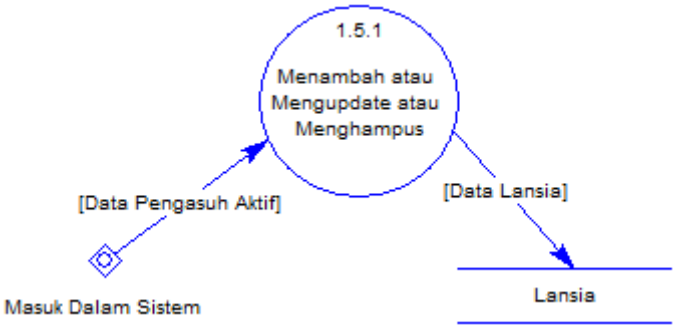
Gambar 3.4 DFD Level 1



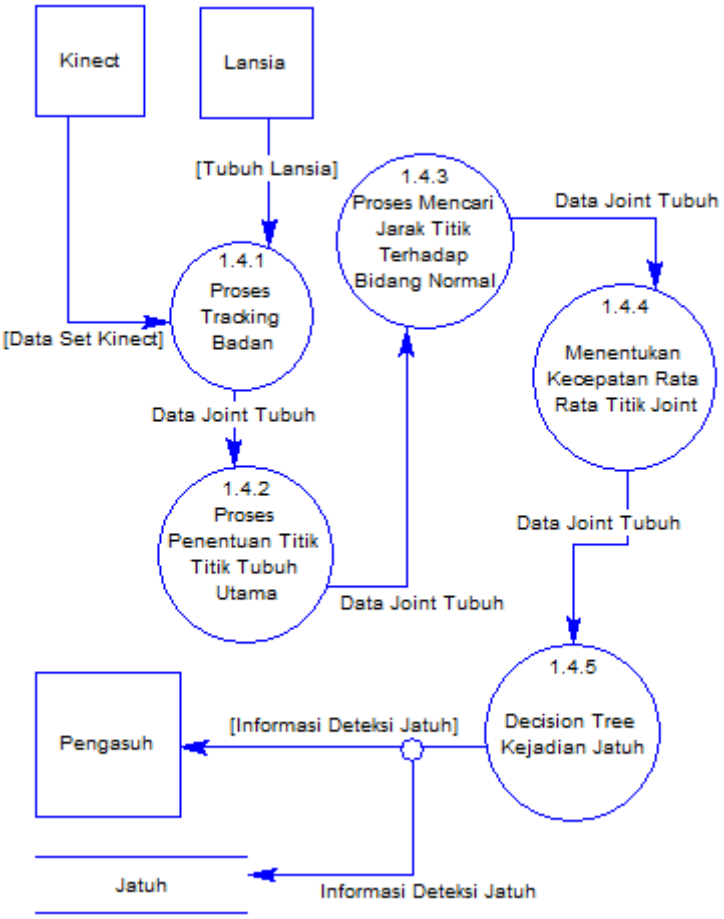
Gambar 3.5 DFD Level 1 untuk proses registrasi

3.1.3.4. DFD Level 2

DFD Level 2 menngambarkan dua dekomposisi dari proses mengelola data lansia dan proses mendeteksi jatuh. Untuk lebih detail dapat dilihat pada Gambar 3.6 untuk dekomposisi mengelola data lansia dan Gambar 3.7 untuk dekomposisi proses pendeteksian jatuh.



Gambar 3.6 Dekomposisi Proses Mengelola Data Lansia



Gambar 3.7 Dekomposisi Proses Mendeteksi Jatuh

3.1.3.5. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional merupakan kebutuhan yang mendukung performa dari sistem yang akan dibangun. Kebutuhan non-fungsional yang dibutuhkan sistem ini antara lain:

a. Sistem akan dikembangkan pada sistem operasi Windows

b. Sistem membutuhkan Kinect for Windows sebagai sensor untuk melakukan Pengawasan deteksi jatuh pada lansia.
c. Sistem menggunakan basis data MySQL dalam menyimpan data pelatihan.

3.1.4. Identifikasi Pengguna

Mengacu pada spesifikasi kebutuhan fungsional yang telah dipaparkan, telah ditentukan pihak-pihak, baik manusia maupun sistem/ perangkat lain yang terlibat secara langsung dengan sistem. Pada perangkat lunak ini, terdapat pengguna utama yaitu pengasuh. Detail tugas dan hak akses pengguna dapat dilihat pada Tabel 3.2 berikut.

Tabel 3.2 Detail Tugas dan Hak Akses Pengguna

Kategori Pengguna	Detail Tugas	Hak Akses ke Aplikasi	Kemampuan Harus Dimiliki
Pengasuh	Menngelola data lansia dan mengawasi dengan mengaktifkan mode pengawasan pendeteksi jatuh dengan Kinect	Harus memiliki akun yang terdaftar pada sistem untuk dapat melakukan tugas dan hak akses	Memiliki kemampuan merawat, mengawasi, lansia dan menjalankan aplikasi.

3.2. Perancangan Sistem

Tahap ini meliputi perancangan basis data, tampilan antarmuka, dan perancangan alur proses penggunaan sistem yang diharapkan dapat memenuhi tujuan dari pengembangan aplikasi ini. Perlu diketahui bahwa aplikasi ini dibangun dalam kondisi lingkungan tertentu, dan dapat dioperasikan dalam lingkungan tertentu pula.

3.2.1. Lingkungan Perancangan

Pada perancangan aplikasi, dibutuhkan spesifikasi lingkungan perancangan tertentu. Spesifikasi perangkat keras dan perangkat lunak yang akan digunakan pada tahap perancangan perangkat lunak seperti yang akan dijelaskan pada Tabel 3.3.

Tabel 3.3 Lingkungan Perancangan Perangkat Lunak

Perangkat Keras (ASUS-N46VM)	Prosesor	Prosesor Intel® Core™ i7-3610QM CPU @ 2.30 GHz (8 CPUs)
	Memori Primer	4096 MB
	Memori sekunder	750 GB
	Display	• NVIDIA GT 630M • Intel(R) HD Graphics 4000
Perangkat Lunak	Sistem Operasi	Windows 10 Pro 64-bit (10.0, Build 10586)
	Perangkat Lunak	• Visual Studio 2015 • Microsoft Word 2016 • Kinect SDK for Windows 2.0 • CorelDraw X8 (64-bit) • Adobe Photoshop CC 2015 • ProccessAnalyst 6.0. • Weka

3.2.2. Deskripsi Proses

Pada subbab ini akan dijelaskan dari setiap proses pada DFD disetiap levelnya.

3.2.2.1. Proses 1.1: Mendaftar Pada Sistem

Proses mendaftar pada sistem dilakukan oleh pengasuh ketika pertama kali menggunakan sistem. Pada proses ini, pengasuh memasukkan data-data yang dibutuhkan dan diminta sistem.

3.2.2.2. Proses 1.2: Masuk Dalam Sistem

Proses masuk dalam sistem dilakukan pengasuh dengan melakukan *login*. Pengasuh diminta memasukkan data-data tertentu sesuai yang telah diisikan ketika registrasi sebelumnya.

3.2.2.3. Proses 1.3: Mengaktifkan Kinect

Proses ini terjadi ketika pengasuh menekan tombol pemicu diaktifkannya Kinect. Pada proses ini sistem akan memanggil Kinect untuk menjadi mode aktif.

3.2.2.4. Proses 1.4: Proses Mendeteksi Jatuh

Proses mendeteksi jatuh terjadi seketika Kinect aktif. Kinect akan mulai menangkap *frame* per *frame* melalui sensor yang dimiliki. Proses ini terpicu dengan aktifnya sensor penangkap pada Kinect. Proses ini terdiri dari banyak subproses didalamnya hingga ditentukannya kejadian jatuh. Setelah mendeteksi jatuh, selanjutnya dilakukan penyampaian informasi telah terjadi jatuh pada pengguna/ pengasuh. Selain itu, informasi jatuh juga akan disimpan pada *database* aplikasi.

3.2.2.5. Proses 1.4.1: Proses Tracking Badan

Pada proses *tracking* badan, tubuh lansia akan di-*track* secara saksama oleh Kinect. Dengan begitu Kinect akan mendeteksi secara *realtime* kemanapun objek badan yang telah ter-*track* akan pergi.

3.2.2.6. Proses 1.4.2: Penentuan Titik-Titik Tubuh Utama

Pada proses ini, titik-titik tubuh yang telah terbaca oleh Kinect akan dikurangi dengan hanya mengambil empat titik utama pada titik *joint* tubuh. Titik-titik yang diambil diantaranya adalah titik kepala, titik tulang belakang, titik bahu kanan, dan titik bahu kiri.

3.2.2.7. Proses 1.4.3: Proses Mencari Jarak Titik Terhadap Bidang Normal

Dalam proses pencarian jarak titik-titik yang telah ditentukan terhadap bidang normal, Kinect akan menentukan dahulu posisi dimensi dari bidang normal yang kemudian setelah itu dapat ditemukan jarak terhadap titik tubuh.

3.2.2.8. Proses 1.4.4: Menentukan Kecepatan Rata-Rata Titik Joint Tubuh

Pada proses ini, dilakukan kalkulasi waktu relatif yang didapat ketika Kinect memroses titik tubuh kemudian mencari kecepatan terhadap jarak dari bidang normal. Setelah mendapat

kecepatan, dapat ditemukan juga kecepatan rata-rata tiap jumlah *frame* yang ditangkap Kinect.

3.2.2.9. Proses 1.4.5: Decision Tree Kejadian Jatuh

Proses ini adalah proses terakhir yang akan dilakukan dalam penentuan kejadian jatuh. Pada proses ini, data-data yang diperoleh dari subproses-subproses Proses mendeteksi jatuh pada *decision tree* yang akhirnya akan menentukan apakah telah terjadi jatuh atau tidak.

3.2.2.10. Proses 1.5: Mengelola Data Lansia

Mengelola data lansia merupakan peroses lain yang dapat dilakukan oleh pengasuh dalam sistem. Mengelola data lansia meliputi menambah, mengubah dan atau menghapus data.

3.2.2.11. Proses 1.5.1: Menambah, atau Mengupdate, atau Menghapus

Pada proses ini, pengasuh bisa melakukan penambahan, pengubahan hingga penghapusan data lansia pada sistem. Dengan mengisikan data lansia baru, pengasuh bisa menambah data baru pada *store* lansia. Kemudian dengan memilih data yang akan dihapus atau diubah maka sistem akan menghapus atau mengubah dan menyimpan di *store* lansia.

3.2.3. Deskripsi Data

Deskripsi data berisikan pemaparan data-data yang dibutuhkan pada proses dalam aplikasi. Pada pemaparannya diberikan deskripsi mengenai semua data-data tersebut. Deskripsi data pada aplikasi pendeteksi jatuh yang dibangun dapat pada Tabel 3.4 berikut.

Tabel 3.4 Deskripsi Data (a)

Nama Data	Deskripsi
Data Registrasi Pengasuh	Data Registrasi Pengasuh adalah data-data yang diisikan oleh pengasuh dan diminta oleh sistem ketika melakukan registrasi pada awal penggunaan aplikasi

Tabel 3.5 Deskripsi Data (b)

Nama Data	Deskripsi
Data *Login* Pengasuh	Data *login* registrasi meliputi *username* dan *password* yang dimasukkan oleh pengasuh ketika hendak mengakses aplikasi
Data Pengasuh Aktif	Data pengasuh aktif adalah data yang aktif ketika pengasuh melakukan akses di dalam aplikasi.
Data Set Kinect	Data perintah untuk mengeset dan mengaktifkan Kinect.
Data Lansia	Data lansia yang merupakan masukan dari pengasuh yang berisi data-data berkaitan lansia dalam sistem.
Tubuh Lansia	Adalah tubuh lansia yang akan menjadi masukan untuk Kinect yang akan ditangkap
Data *Joint* Tubuh	Data *joint* tubuh merupakan kelas berisi data per *frame* yang nantinya memiliki atribut-atribut hasil ekstraksi dan yang nantinya akan dimasukkan ke *decision tree*.
Informasi Deteksi Jatuh	Informasi Deteksi Jatuh adalah informasi yang akan didapat oleh pengasuh ketika aplikasi berhasil mengenali terjadinya kejadian jatuh.

3.2.4. Perancangan Kamus Data

Pada tahap ini akan dijelaskan data yang digunakan pada tiap proses pada *Data Flow Diagram* yang telah dipaparkan pada subbab 3.1 sebelumnya. Rincian yang ditampilkan pada kamus data adalah nama arus data, bentuk data, arus data, penjelasan, hingga struktur data.

3.2.4.1. Data Registrasi Pengasuh

Arus data registrasi pengasuh merupakan data-data yang harus dimasukan ketika pertama kali ketika mengakses aplikasi. Data adalah masukan berupa teks yang mencakup data-data berkaitan dengan profil pengasuh. Data ini harus dimasukkan untuk mendapatkan akses ke dalam aplikasi. Untuk lebih detail dapat dilihat pada Tabel 3.6 di bawah.

Tabel 3.6 Deskripsi Data Registrasi Pengasuh

<table>
<tr><th colspan="2">Kamus Data</th></tr>
<tr><td>Nama Arus Data</td><td>Data Registrasi Pengasuh</td></tr>
<tr><td>Alias</td><td>-</td></tr>
<tr><td>Bentuk Data</td><td>Persisten</td></tr>
<tr><td>Arus Data</td><td>1. Pengasuh → Proses 1.1</td></tr>
<tr><td>Penjelasan</td><td>Data Registrasi Pengasuh adalah data masukkan pengasuh untuk registrasi ketika pertama kali menggunakan aplikasi</td></tr>
<tr><td>Struktur Data</td><td>
<table>
<tr><th>Item</th><th>Tipe</th><th>Keterangan</th></tr>
<tr><td>pengasuh_id</td><td>Varchar(10)</td><td>Diisi user</td></tr>
<tr><td>pengasuh_nama</td><td>Varchar(200)</td><td>Diisi user</td></tr>
<tr><td>pengasuh_email</td><td>Varchar(100)</td><td>Diisi user</td></tr>
<tr><td>pengasuh_password</td><td>Varchar(20)</td><td>Diisi user</td></tr>
<tr><td>pengasuh_tmplahir</td><td>Varchar(100)</td><td>Diisi user</td></tr>
<tr><td>pengasuh_tgllahir</td><td>Date</td><td>Diisi user</td></tr>
<tr><td>pengasuh_alamat</td><td>Varchar(300)</td><td>Diisi user</td></tr>
</table>
</td></tr>
</table>

3.2.4.2. Data Login Pengasuh

Data *login* adalah data yang harus dimasukkan setelah berhasil registrasi untuk mengakses aplikasi. Data *login* merupakan sebagian data registrasi. Penjelasan lebih detail dapat dilihat pada Tabel 3.7 dan Tabel 3.8.

Tabel 3.7 Deskripsi Data Login Pengasuh (a)

Kamus Data	
Nama Arus Data	Data *Login* Pengasuh
Alias	-
Bentuk Data	Persisten
Arus Data	1. Pengasuh → Proses 1.2
Penjelasan	Data *login* adalah data masukan dari pengasuh yang telah terdaftar/ registrasi pada sistem

Tabel 3.8 Deskripsi Data Login Pengasuh (b)

Kamus Data	
Struktur Data	

Item	Tipe	Keterangan
pengasuh_id	Varchar(10)	Diisi *user*
pengasuh_password	Varchar(20)	Diisi *user*

3.2.4.3. Data Pengasuh Aktif

Data Pengasuh Aktif akan diberikan sistem kepada pengasuh yang telah *login* ke dalam aplikasi sesuai pada Tabel 3.9.

Tabel 3.9 Deskripsi Data Pengasuh

Kamus Data	
Nama Arus Data	Data Pengasuh Aktif
Alias	-
Bentuk Data	Persisten
Arus Data	1. Proses 1.2 → Proses 1.3 2. Proses 1.2 → Proses 1.5
Penjelasan	Pengasuh yang telah *login* dapat masuk ke dalam sistem dengan data aktif yang memberikan *session* pada pengasuh untuk mengakses aplikasi
Struktur Data	

Item	Tipe	Keterangan
pengasuh_id	Varchar(10)	Ditentukan sistem

3.2.4.4. Data Set Kinect

Data Set Kinect akan diberikan sistem ketika pengasuh memberikan aksi menekan tombol yang akan memicu sistem untuk mengaktifkan Kinect yang ditunjukkan Tabel 3.10 dan Tabel 3.11.

Tabel 3.10 Deskripsi Data Set Kinect (a)

Kamus Data	
Nama Arus Data	Data Set Kinect
Alias	-

Tabel 3.11 Deskripsi Data Set Kinect (b)

<table>
<tr><th colspan="2">Kamus Data</th></tr>
<tr><td>Bentuk Data</td><td>Persisten</td></tr>
<tr><td>Arus Data</td><td>1. Proses 1.3 → Kinect
2. Kinect → Proses 1.4</td></tr>
<tr><td>Penjelasan</td><td>1. Pada arus data ini, pengasuh akan menekan tombol pemicu yang menyebabkan Kinect akan
2. Kemudian sesaat setelah Kinect aktif, aplikasi akan memroses tangkapan Kinect dan menjalankan proses pendeteksian jatuh</td></tr>
<tr><td>Struktur Data</td><td>
<table>
<tr><th>Item</th><th>Tipe</th><th>Keterangan</th></tr>
<tr><td>statusKinect</td><td>bool</td><td>Berubah sesuai kejadian pada pemicu</td></tr>
</table>
</td></tr>
</table>

3.2.4.5. Data Lansia

Data Lansia adalah masukan dari Pengasuh untuk melakukan pengelolaan data informasi berkaitan dengan lansia yang ada dalam *database* aplikasi. Pengelolaan data tersebut meliputi penambahan, pengubahan, dan penghapusan data. Untuk lebih jelasnya mengenai deskripsi data ini, akan dijelaskan lebih lanjut pada Tabel 3.12 dan Tabel 3.13.

Tabel 3.12 Deskripsi Data Lansia (a)

Kamus Data	
Nama Arus Data	Data Lansia
Alias	-
Bentuk Data	Persisten
Arus Data	1. Proses 1.5 → *Storage* Lansia
Penjelasan	Pengasuh dapat melakukan pengelolaan terhadap data lansia dengan memasukkan data-data masukan yang diminta terkait lansia.

Tabel 3.13 Deskripsi Data Lansia (b)

Kamus Data	
Struktur Data	

Item	Tipe	Keterangan
lansia_id	Varchar(10)	Diisi *user*
lansia_nama	Varchar(200)	Diisi *user*
lansia_tmplahir	Varchar(100)	Diisi *user*
lansia_tgllahir	Date	Diisi *user*
lansia_deskr	Varchar(500)	Diisi *user*

3.2.4.6. Tubuh Lansia

Tubuh lansia adalah objek tubuh yang nantinya akan dideteksi oleh Kinect Sebelum aplikasi nantinya akan mendeteksi gerakan tubuh tersebut untuk mengklasifikasikan gerakan tersebut jatuh atau tidak. Untuk lebih detail dan jelas dapat dilihat pada Tabel 3.14 berikut.

Tabel 3.14 Deskripsi Data Tubuh Lansia

Kamus Data	
Nama Arus Data	Tubuh Lansia
Alias	-
Bentuk Data	Persisten
Arus Data	1. Lansia → Proses 1.4
Penjelasan	Dengan diaktifkannya mode pengawasan, Kinect akan mulai dengan meng-*capture* diikuti mendeteksi objek tubuh manusia. Objek tubuh ini merupakan yang nantinya akan dideteksi apakah jatuh atau tidak.

3.2.4.7. Data Joint Tubuh

Setelah Kinect berhasil menangkap dan mendeteksi objek tubuh, akan dilakukan proses selanjutnya terhadap tangkapan Kinect untuk mengambil data-data yang dibutuhkan untuk pendeteksian jatuh. Beberapa diantaranya adalah titik-titik tubuh tiap *frame* yang ditangkap oleh Kinect. Untuk lebih jelasnya, dapat dilihat pada Tabel 3.15 berikut.

Tabel 3.15 Deskripsi Data Joint Tubuh

<table>
<tr><th colspan="2">Kamus Data</th></tr>
<tr><td>Nama Arus Data</td><td>Data Joint Tubuh</td></tr>
<tr><td>Alias</td><td>-</td></tr>
<tr><td>Bentuk Data</td><td>Persisten</td></tr>
<tr><td>Arus Data</td><td>Proses 1.4.1 → Proses 1.4.2 → Proses 1.4.3 → Proses 1.4.4 → Proses 1.4.5</td></tr>
<tr><td>Penjelasan</td><td>Data joint tubuh pada dasarnya adalah sebuah objek yang memiliki atribut-atribut tertentu yang dibutuhkan untuk perhitungan dan pemrosesan lebih jauh dalam mendeteksi jatuh. Joint tubuh ini nantinya akan diubah menjadi objek titik. Atribut tersebut antara lain adalah posisi titik joint, jarak, hingga kecepatan perubahan jarak. Data ini yang pada akhirnya akan di proses pada decision tree guna mengklasifikasikan apakah terdeteksi jatuh atau tidak.</td></tr>
<tr><td>Struktur Data</td><td>
<table>
<tr><th>Item</th><th>Tipe</th><th>Keterangan</th></tr>
<tr><td>Posisi</td><td>Vector3D</td><td>-</td></tr>
<tr><td>Jarak sebelum</td><td>Float</td><td>-</td></tr>
<tr><td>Jarak sesudah</td><td>Float</td><td>-</td></tr>
<tr><td>Kecepatan</td><td>Float</td><td>-</td></tr>
<tr><td>Kecepatan Rata-rata</td><td>Float</td><td>-</td></tr>
</table>
</td></tr>
</table>

3.2.4.8. Informasi Deteksi Jatuh

Setelah data *joint* masuk ke *decision tree,* akan ada proses ketika terdeteksi jatuh dimana aplikasi memberikan notifikasi *popup* yang dapat dilihat pada Tabel 3.16 dan Tabel 3.17.

Tabel 3.16 Deskripsi Data Informasi Deteksi Jatuh (a)

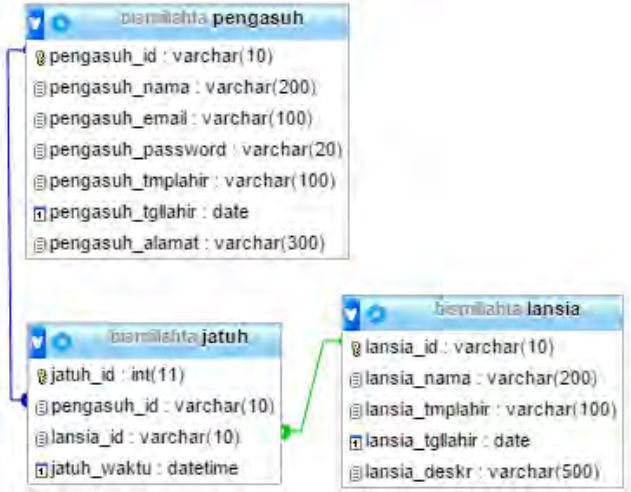
Kamus Data	
Nama Arus Data	Informasi Deteksi Jaruh
Alias	-

Tabel 3.17 Deskripsi Data Informasi Deteksi Jatuh (b)

Kamus Data			
Bentuk Data	Persisten		
Arus Data	Proses 1.4.5 → Pengasuh Proses 1.4.5 → *Storage* Jatuh		
Penjelasan	Informasi mengenai jatuh yang menyimpan data masukan dari sistem seketika jatuh terdeteksi.		
Struktur Data			
	Item	Tipe	Keterangan
	jatuh_id	Varchar(10)	Diisi sistem
	jatuh_waktu	Varchar(200)	Diisi sistem

3.2.5. Perancangan Basis Data

Pada subbab ini dijelaskan mengenai perancangan basis data yang dalam hal ini digunakan untuk menyimpan data diri partisipan beserta rekam skor yang diperolehnya selama menggunakan aplikasi ini. Gambaran perancangan basis data dapat dilihat pada Gambar 3.14 berikut.



Gambar 3.8 Rancangan Basis Data Aplikasi

3.2.5.1. Rancangan Tabel Pengasuh

Tabel pengasuh digunakan untuk menyimpan data prngguna utama yaitu pengasuh. Tabel pengasuh memiliki relasi ke tabel lansia dengan hubungan *many to many* yang meneybab

kan terbuatnya tabel baru yaitu tabel jatuh. Data atribut pada tabel pengasuh dapat dilihat pada Tabel 3.18

Tabel 3.18 Atribut Tabel Pengasuh

Nama Kolom	Tipe	Keterangan
pengasuh_id	Varchar(10)	*Primary Key* tabel pengasuh
pengasuh_nama	Varchar(200)	Nama pengasuh
pengasuh_email	Varchar(100)	*Email* pengasuh
pengasuh_password	Varchar(20)	*Password* akses pengasuh
pengasuh_tmplahir	Varchar(100)	Tempat lahir pengasuh
pengasuh_tgllahir	Date	Tanggal lahir pengasuh
pengasuh_alamat	Varchar(300)	Alamat pengasuh

3.2.5.2. Rancangan Tabel Lansia

Tabel lansia hampir sama dengan tabel pengasuh. Sama-sama merupakan tabel master, dimana data pada tabel lansia diinput oleh pengasuh. Begitu pula dengan relasinya yang saling berelasi dengan tabel pengasuh menghasilkan tabel jatuh. Detail dari tabel lansia dapat dilihat pada Tabel 3.19

Tabel 3.19 Atribut Detail Tabel Lansia

Nama Kolom	Tipe	Keterangan
lansia_id	Varchar(10)	*Primary Key* tabel lansia
lansia_nama	Varchar(200)	Nama lengkap lansia
lansia_tmplahir	Varchar(100)	Tempat lahir lansia
lansia_tgllahir	Date	Tanggal lahir lansia
lansia_deskr	Varchar(500)	Deskripsi singkat mengenai lansia

3.2.5.3. Rancangan Tabel Jatuh

Tabel jatuh adalah tabel yang menyimpan data jatuh sesaat ketika terdeteksi. Tabel ini merupakan hasil dari relasi *many to*

many antara tabel pengasuh dengan tabel lansia. Atribut detail tabel jatuh dapat dilihat di Tabel 3.20.

Tabel 3.20 Atribut Tabel Jatuh

Nama Kolom	Tipe	Keterangan
jatuh_id	Varchar(10)	Diisi sistem
pengasuh_id	Varchar(10)	*Foreign Key* dari tabel pengasuh
Lansia_id	Varchar(10)	*Foreign Key* dari tabel lansia
jatuh_waktu	Varchar(200)	Waktu sekarang ketika jatuh terdeteksi

3.2.6. Perancangan Antarmuka

Pada subbab ini akan dijelaskan secara detail rancangan antarmuka aplikasi pendeteksi jatuh yang ditujukan untuk *platform* Windows.

3.2.6.1. Jendela Login

Jendela *Login* adalah antarmuka pertama yang muncul ketika aplikasi berjalan. Pada antarmuka ini, pengguna yaitu pengasuh diminta memasukkan *username* dan *password* untuk dapat mengakses aplikasi. Rancangan antarmuka *login* secara detail dapat dilihat pada Gambar 3.9.

Gambar 3.9 Rancangan Jendela Login

Berikut penjelasan komponen antarmuka pada gambar di atas :

1. Logo aplikasi;
2. Tulisan deskripsi singkat mengenai *headline* aplikasi;
3. Kotak masukan *username*;
4. Kotak masukan *password*;
5. Tombol untuk masuk menuju antarmuka utama;
6. Tombol untuk registrasi, menuju antarmuka registrasi.

3.2.6.2. Jendela Registrasi

Jendela registrasi adalah antarmuka bagi pengguna untuk melakukan registrasi pendaftaran pada sistem. Sebelum bisa masuk *login* pada sistem, pengguna atau pengasuh diharuskan untuk registrasi. Jendela ini bisa terpanggil melalui tombol pada antarmuka *login* yang terpicu dengan ditekan. Penjelasan detail antarmuka registrasi dapat dilihat pada

Gambar 3.10 Rancangan Jendela Registrasi

Berikut penjelasan komponen antarmuka registrasi pada gambar di atas :

1. Logo aplikasi;
2. Tulisan ‘registrasi’ dan deskripsi singkatnya;
3. Kotak masukan nama lengkap pengasuh;

4. Kotak masukan tempat lahir pengasuh;
5. Kotak masukan tanggal lahir pengasuh;
6. Kotak masukan alamat pengasuh;
7. Kotak masukan *username* pengasuh;
8. Kotak masukan *email* pengasuh;
9. Kotak masukan *password* pengasuh;
10. Kotak masukan konfirmasi *password* pengasuh;
11. Tombol untuk registrasi/ daftar.

3.2.6.3. Jendela Utama – Tab "Tentang"

Jendela utama adalah antarmuka yang akan bisa diakses ketika seorang pengasuh telah masuk melalui antarmuka *login*. Dengan antarmuka ini, pengasuh bisa mengakses fitur fitur pada aplikasi yang terbagi dalam *tab*. *Tab* pertama adalah *tab* "Tentang" yang berisi informasi singkat sekilas aplikasi. Penjelasan komponen antarmuka *tab* "Tentang" adalah seperti ditunjukkan pada Gambar 3.11.

Gambar 3.11 Rancangan Jendela Utama - Tab Tentang

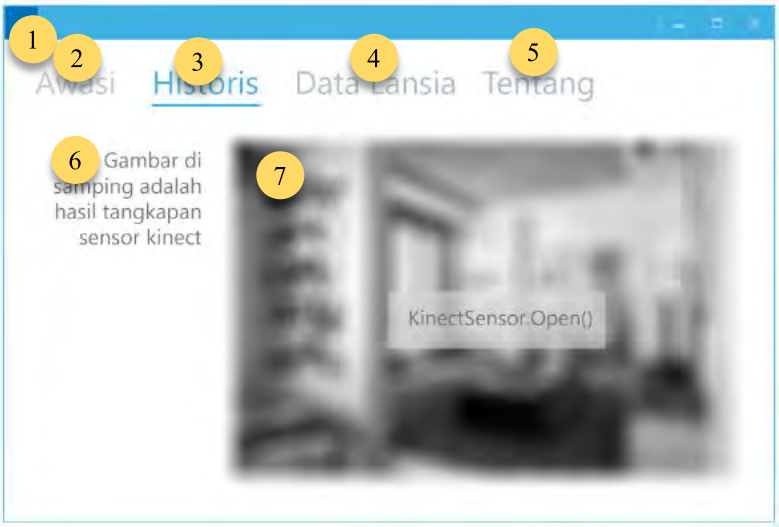
Berikut penjelasan komponen antarmuka utama *tab* "Tentang "pada gambar di atas :

1. Logo aplikasi;

2. Tab “Awasi” untuk berpindah *tab* ke *tab* “Awasi”;
3. Tab “Historis” untuk berpindah *tab* ke *tab* “Historis”;
4. Tab “Data Lansia” untuk berpindah *tab* ke *tab* “Data Lansia”;
5. Tab “Tentang” untuk berpindah *tab* ke *tab* “Tentang”;
6. Tulisan deskripsi singkat mengenai aplikasi yang merupakan konten pada *tab* “Tentang”;

3.2.6.4. Jendela Utama – Tab “Awasi”

Jendela utama – *tab* “Awasi” adalah antarmuka yang memberikan *view* pada pengguna/ pengasuh berupa tangkapan Kinect secara *realtime*. Penjelasan secara terperinci dapat dilihat pada Gambar 3.12.



Gambar 3.12 Rancangan Jendela - Tab Awasi

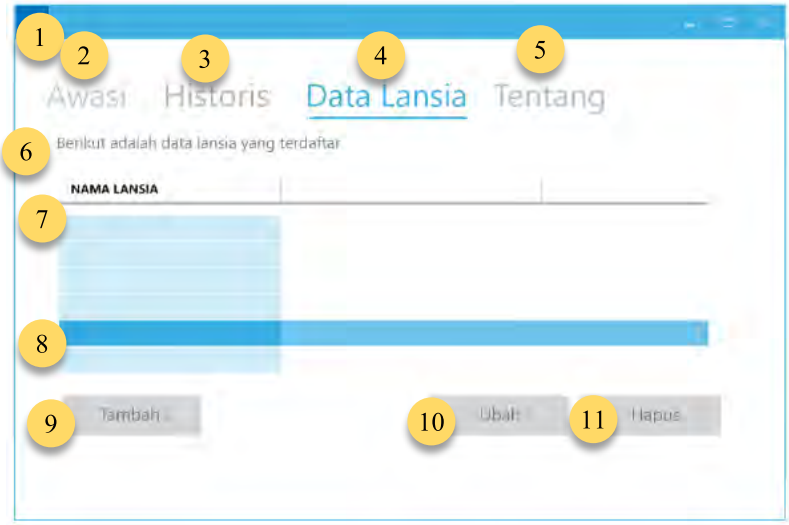
Berikut penjelasan komponen antarmuka utama *tab* “Awasi” pada gambar di atas :

1. Logo aplikasi;
2. Tab “Awasi” untuk berpindah *tab* ke *tab* “Awasi”;
3. Tab “Historis” untuk berpindah *tab* ke *tab* “Historis”;

4. Tab “Data Lansia” untuk berpindah *tab* ke *tab* “Data Lansia”;
5. Tab “Tentang” untuk berpindah *tab* ke *tab* “Tentang”;
6. Tulisan deskripsi singkat tentang *tab* “Awasi”;
7. Gambar hasil tangkapan Kinect secara *realtime*.

3.2.6.5. Jendela Utama – Tab “Data Lansia”

Jendela utama *tab* “Data Lansia” diperuntukkan pengasuh untuk mengakses *database* yang memuat data lansia. Pada antarmuka ini, pengasuh bisa melakukan pengelolaan data. Secara detail dapat dilihat pada Gambar 3.13.



Gambar 3.13 Rancangan Jendela Utama Tab Data Lansia

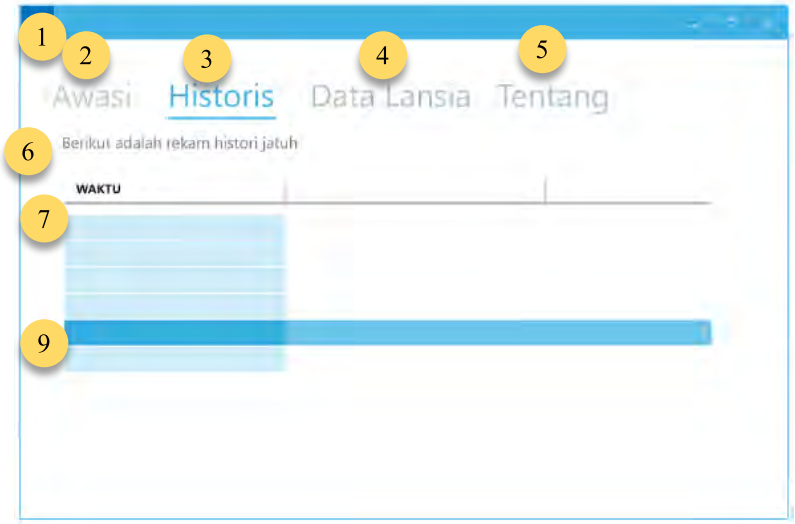
Berikut penjelasan komponen antarmuka utama *tab* “Data Lansia” pada gambar di atas :

1. Logo aplikasi;
2. Tab “Awasi” untuk berpindah *tab* ke *tab* “Awasi”;
3. Tab “Historis” untuk berpindah *tab* ke *tab* “Historis”;
4. Tab “Data Lansia” untuk berpindah *tab* ke *tab* “Data Lansia”;
5. Tab “Tentang” untuk berpindah *tab* ke *tab* “Tentang”;

6. Tulisan deskripsi singkat tentang *tab* "Data Lansia";
7. Tabel yang memuat data lansia dari *database*.
8. Data lansia yang sedang terpilih.
9. Tombol tambah, untuk menambah lansia;
10. Tombol ubah untuk mengubah data lansia terpilih;
11. Tombol hapus untuk menghapus data lansia terpilih.

3.2.6.6. Jendela Utama – Tab "Histori"

Amtarmuka utama *tab* "Histori" adalah antarmuka untuk menampilkan data jatuh yang berhasil terdeteksi dan tersimpan di *database*. Penjelasan secara terperinci dapat dilihat pada Gambar 3.14.



Gambar 3.14 Rancangan Jendela Utama Tab Histori

Berikut penjelasan komponen antarmuka utama *tab* "Data Lansia" pada gambar di atas :

1. Logo aplikasi;
2. Tab "Awasi" untuk berpindah *tab* ke *tab* "Awasi";
3. Tab "Historis" untuk berpindah *tab* ke *tab* "Historis";
4. Tab "Data Lansia" untuk berpindah *tab* ke *tab* "Data Lansia";

5. Tab “Tentang” untuk berpindah *tab* ke *tab* “Tentang”;
6. Tulisan deskripsi singkat tentang *tab* “Histori”;
7. Tabel yang memuat data jatuh dari *database*.
8. Data jatuh yang sedang terpilih.

3.2.7. Perancangan Algoritma Decision Tree Sebagai pendeteksi Jatuh

Pada tahap ini akan dijelaskan mengenai rancangan algoritma *decision tree* pendeteksi jatuh yang akan digunakan dalam pengimplementasian aplikasi pendeteksi jatuh. Secara umum, algoritma ini akan mengklasifikasikan apakah *frame* hasil tangkapan Kinect yang berisi data-data tubuh merupakan jatuh atau tidak. Untuk membangun *decision tree* yang diinginkan, beberapa langkah yang harus dilakukan adalah sebagai berikut:

3.2.7.1. Ektraksi Fitur Aktivitas Jatuh

Tahap ekstraksi fitur adalah tahap dimana pengamatan terhadap orang yang mengalami jatuh dilakukan. Dengan melihat pola gerakan jatuh, dapat ditentukan atribut-atribut unik dari gerakan jatuh dibandingkan gerakan lainnya. Dimulai dengan kemampuan Kinect beserta SDK nya untuk mendeteksi tubuh manusia, dapat diambil titik-titik *joint* pada tubuh manusia mulai dari kepala, pundak, sampai dengan kaki. Selain itu, Kinect juga mampu mendeteksi bidang normal dimana tubuh manusia yang terdeteksi itu berdiri. Dengan demikian, Fitur-fitur yang dapat ditentukan tersebut antara lain adalah:

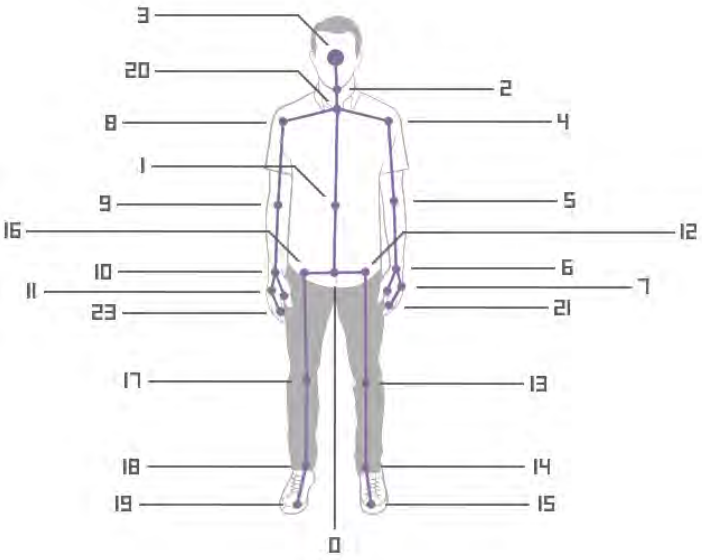
3.2.7.1.1. Penentuan titik-titik joint sebagai atribut

Penentuan titik sangat berpengaruh terhadap atribut yang nantinya akan diekstraksi fiturnya pada tahap pengumpulan *dataset*. Pada bab sebelumnya, telah dijelaskan bahwa Kinect v2 mampu menangkap dan mendeteksi 26 titik *joint* pada tubuh seperti ditunjukkan pada Gambar 3.15. Dengan mengamati aktivitas kejadian jatuh akan diketahui bahwa tidak semua titik tersebut digunakan dalam membedakan jatuh dan tidak jatuh. Sehingga perlu dilakukan penentuan titik-titik tersebut. Penentuan titik *joint* pada tubuh didasarkan pada *information gain* paling

besar yang nantinya memengaruhi ketika melakukan pengumpulan *dataset* dan pengelompokan kelas.

Ternyata, didapatkan titik-titik berjumlah empat dengan *gain information* yang cukup tinggi. Titik-titik yang dipilih tersebut diantaranya adalah:

- Kepala (nomor 3 pada Gambar 3.15)
- Tulang Belakang (nomor 20 pada Gambar 3.15)
- Bahu Kiri (nomor 4 pada Gambar 3.15)
- Bahu Kanan (nomor 8 pada Gambar 3.15)



Gambar 3.15 Titik-titik Joint Tubuh pada Kinect V2

3.2.7.1.2. Jarak titik joint terhadap bidang normal

Penentuan jarak ditentukan pada bidang Cartesian, dimana terdiri dari sumbu x, y, z. Dengan menemukan jarak titik *joint* ke bidang normal tiap *frame*, dapat ditemukan selisih jarak dari *frame* ke-i dengan *frame* ke-i + 1. Jarak tersebut dapat dikalkulasikan sesuai pada persamaan 3.1

$$d = \frac{A_x + B_y + C_z + D}{\sqrt{A^2 + B^2 + C^2}} \quad (3.1)$$

Dengan keterangan sebagai berikut:

- d = jarak pada *frame* ke-i
- A_x= konstanta bidang A normal × X_i
- B_x = konstanta bidang B normal × Y_i
- C_x = konstanta bidang C normal × Z_i
- D = konstanta bidang D normal

3.2.7.1.3. Kecepatan titik joint tiap frame

Pada fitur sebelumnya, ketika sudah ditemukan selisih jarak antara *frame,* dapat ditentukan kecepatan perubahan jarak tersebut. Dengan mencari selisih jarak dan mengambil waktu relatif tiap penangkapan titik *joint* pada tiap *frame* tangkapan Kinect, akan ditemukan kecepatan perubahan jarak tiap titik seperti tercantum pada Persamaan 3.2.

$$v_i = \frac{d_{i+1} - d_i}{t_{i+1} - t_i} \quad (3.2)$$

t adalah waktu relatif yang diambil ketika Kinect menangkap titik *Joint* tiap *frame* nya. Waktu tersebut dalam milisekon, sehingga nantinya perlu dibagi dengan 1000.

3.2.7.1.4. Kecepatan rata-rata titik joint pada N frame

Setelah mendapatkan kecepatan tiap *joint* pada tiap *frame*, perlu dicari kecepatan rata-rata tiap *joint* yang terjadi pada *frame* sejumlah N. Kecepatan rata-rata didapat dengan total v dari *frame* ke i sampai ke N-1 dibagi dengan total waktu tiap *frame* tersebut Lebih detail dapat dilihat pada Persamaan 3.3 [9].

$$v_{avg} = \frac{1000}{N-1} \sum_{i=1}^{N-1} \frac{d_{i+1} - d_i}{t_{i+1} - t_i} \quad (3.3)$$

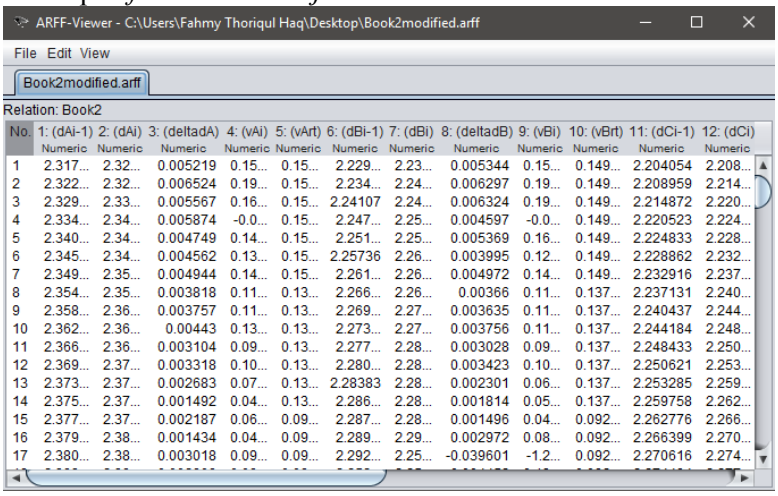
Setelah mendapat kecepatan rata-rata tiap *joint*, kemudian perlu mencari kecepatan rata-rata dari semua *joint.* Seperti dapat dilihat pada persamaan 3.4 [9].

$$v_{avg} = \frac{1}{n}\sum_{i=1}^{n} v_{avg.i} \qquad (3.4)$$

Nilai n di atas adalah jumlah *joint* yang ditentukan sebagai titik dengan *gain information* tertinggi dibandingkan dengan titik *joint* lain pada tubuh.

3.2.7.2. Pengumpulan Dataset

Tahap ini adalah dimana data hasil ekstraksi fitur dikumpulkan yang nantinya akan digunakan untuk *training* membangun *decision tree*. Fitur-fitur yang dimaksud merupakan fitur-fitur yang telah dijelaskan pada subbab 3.2.7.1 sebelumnya, termasuk diantaranya adalah jarak titik *joint* dengan bidang normal, kecepatan tiap titik *joint,* sampai dengan kecepatan rata-rata tiap N *frame* dari titik *joint.*



No.	1: (dAi-1)	2: (dAi)	3: (deltadA)	4: (vAi)	5: (vArt)	6: (dBi-1)	7: (dBi)	8: (deltadB)	9: (vBi)	10: (vBrt)	11: (dCi-1)	12: (dCi)
	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric
1	2.317...	2.32...	0.005219	0.15...	0.15...	2.229...	2.23...	0.005344	0.15...	0.149...	2.204054	2.208...
2	2.322...	2.32...	0.006524	0.19...	0.15...	2.234...	2.24...	0.006297	0.19...	0.149...	2.208959	2.214...
3	2.329...	2.33...	0.005567	0.16...	0.15...	2.24107	2.24...	0.006324	0.19...	0.149...	2.214872	2.220...
4	2.334...	2.34...	0.005874	-0.0...	0.15...	2.247...	2.25...	0.004597	-0.0...	0.149...	2.220523	2.224...
5	2.340...	2.34...	0.004749	0.14...	0.15...	2.251...	2.25...	0.005369	0.16...	0.149...	2.224833	2.228...
6	2.345...	2.34...	0.004562	0.13...	0.15...	2.25736	2.26...	0.003995	0.12...	0.149...	2.228862	2.232...
7	2.349...	2.35...	0.004944	0.14...	0.15...	2.261...	2.26...	0.004972	0.14...	0.149...	2.232916	2.237...
8	2.354...	2.35...	0.003818	0.11...	0.13...	2.266...	2.26...	0.00366	0.11...	0.137...	2.237131	2.240...
9	2.358...	2.36...	0.003757	0.11...	0.13...	2.269...	2.27...	0.003635	0.11...	0.137...	2.240437	2.244...
10	2.362...	2.36...	0.00443	0.13...	0.13...	2.273...	2.27...	0.003756	0.11...	0.137...	2.244184	2.248...
11	2.366...	2.36...	0.003104	0.09...	0.13...	2.277...	2.28...	0.003028	0.09...	0.137...	2.248433	2.250...
12	2.369...	2.37...	0.003318	0.10...	0.13...	2.280...	2.28...	0.003423	0.10...	0.137...	2.250621	2.253...
13	2.373...	2.37...	0.002683	0.07...	0.13...	2.28383	2.28...	0.002301	0.06...	0.137...	2.253285	2.259...
14	2.375...	2.37...	0.001492	0.04...	0.13...	2.286...	2.28...	0.001814	0.05...	0.137...	2.259758	2.262...
15	2.377...	2.37...	0.002187	0.06...	0.09...	2.287...	2.28...	0.001496	0.04...	0.092...	2.262776	2.266...
16	2.379...	2.38...	0.001434	0.04...	0.09...	2.289...	2.29...	0.002972	0.08...	0.092...	2.266399	2.270...
17	2.380...	2.38...	0.003018	0.09...	0.09...	2.292...	2.25...	-0.039601	-1.2...	0.092...	2.270616	2.274...

Gambar 3.16 *Dataset* Untuk Training

Dalam prosesnya, seperti dapat dilihat pada Gambar 3.16 tahap ini melibatkan Weka sebagai aplikasi yang mampu membantu penulis dalam membangun *decision tree* pendeteksi jatuh. Pada tahap awal, dibuat skenario aktivitas, baik jatuh maupun tidak jatuh, kemudian pengumpulan *dataset* dimulai dengan melakukan peragaan skenario tersebut beserta penyimpanan data-data atribut yang berhasil ditangkap Kinect.

Dataset ini nantinya berguna untuk menemukan *threshold* dan batasan nilai untuk menentukan *step decision* pada *decision tree.* Proses pengumpulan *dataset* dimulai dengan mensimulasikan jatuh dengan berbagai skenario dan menyimpan hasil yang didapat dalam bentuk ektraksi fitur ke dalam sebuah *file*. Tiap ekstraksi fitur yang didapat akan diberi sebuah label yang menandakan atribut-atribut. Setelah itu, diperlukan penambahan satu label sebagai kelas untuk klasifikasi. Dalam kasus jatuh ini, kelas klasifikasi hanya terbagi menjadi dua, yaitu terdeteksi jatuh dan tidak terdeteksi. Pada tahap ini dihasilkan sekitar 5200 lebih *frame dataset* dari proses pengumpulan. Dari jumlah ini, terdiri dari sekitar 329 *frame* yang memiliki label "jatuh" dan sisanya, ssekitar 4902 *frame* memiliki label "tidak jatuh".

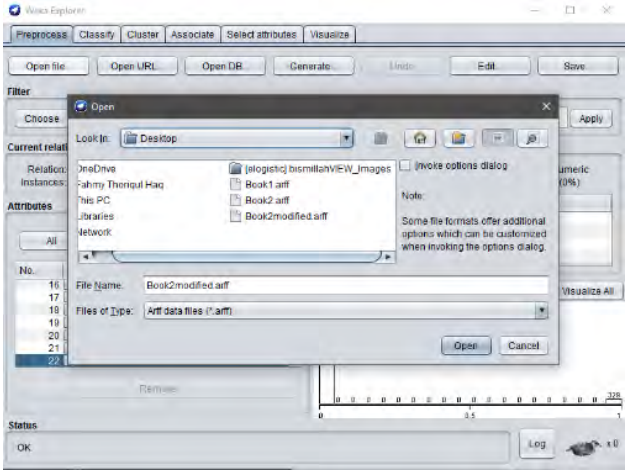
3.2.7.3. Membangun Decision Tree

Tahap ini secara garis besar adalah penjelasan bagaimana proses yang harus dilakukan membangun *tree* yang diinginkan menggunakan Weka. langkah-langkah pada tahap ini adalah sebagai berikut:

1. Buka aplikasi Weka, maka akan muncul jendela Weka seperti terlihat pada Gambar 3.17.

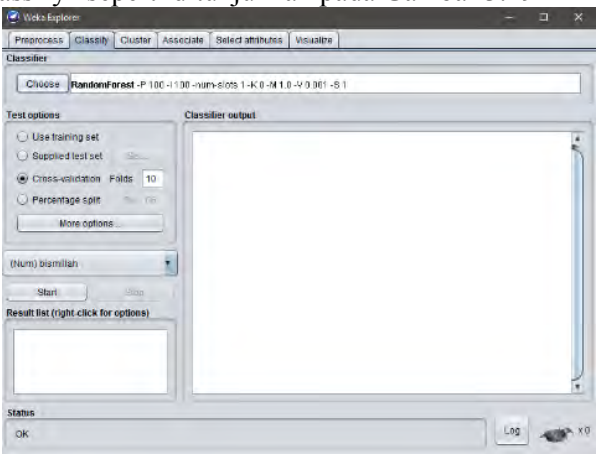
Gambar 3.17 Jendela Awal Aplikasi Weka

2. Klik "Explorer" pada *menu* "Applications" di bagian kanan jendela awal Weka.
3. Setelah jendela *explorer* ditampilkan, buka *file dataset* pada Weka *explorer* seperti ditunjukkan pada Gambar 3.18



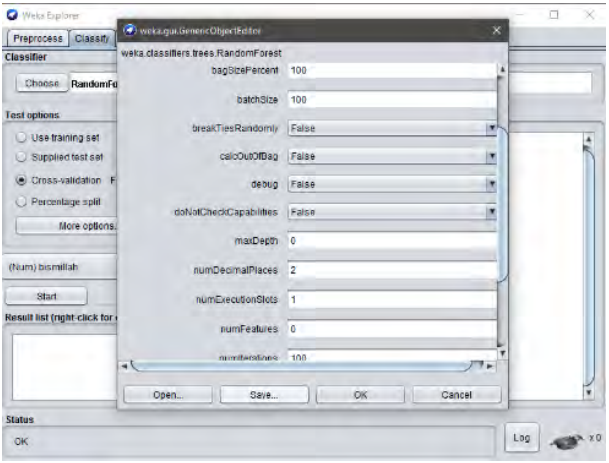
Gambar 3.18 Membuka File *Dataset* pada Weka Explorer

4. Setelah *dataset* termuat pada aplikasi, pindah menuju *tab* "Classify" seperti ditunjukkan pada Gambar 3.19



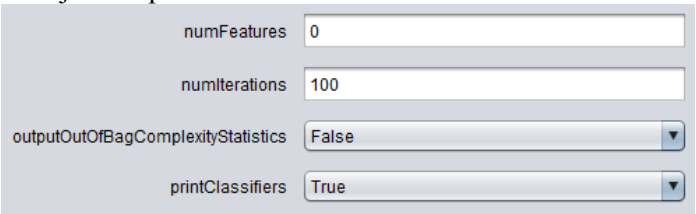
Gambar 3.19 Tab Classify Weka

5. Pada *tab* “Classify” di bagian “Classifier” tepatnya, pilih metode dan ubah menjadi *trees* RandomForest seperti diperlihatkan pada Gambar 3.19.
6. Kemudian klik pada *textbox* “RandomForest” untuk mengeluarkan jendelea opsi lanjut pada RandomForest yang akan diaplikasikan yang ditunjukkan pada Gambar 3.20 berikut.



Gambar 3.20 Jendela Opsi Lebih Untuk Classifier

7. Ubah opsi “printClassifier” ke *true* untuk mencetak *tree* yang nantinya dihasilkan RandomForest ini. Seperti ditunjukkan pada Gambar 3.21

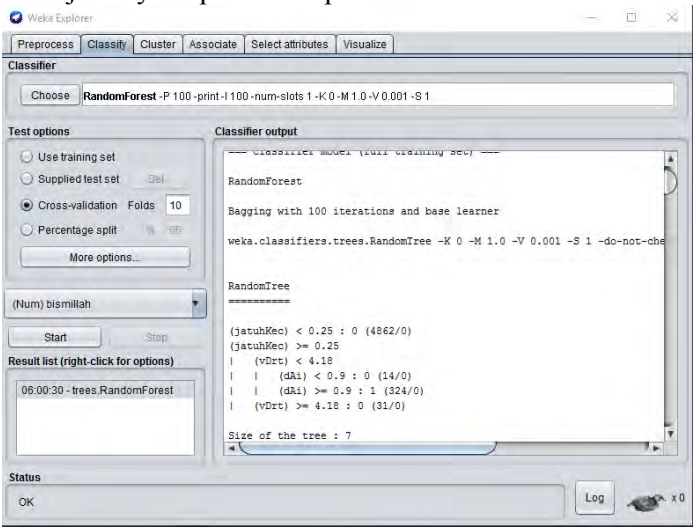


Gambar 3.21 Ubah opsi "printClassifier" Menjadi True

8. Di bagian “Test Options” tentukan label atribut yang menjadi kelas pengklasifikasian. Pada kasus kali ini, dipilihlah atribut “bismillah” yang merupakan atribut

dengan nilai 1 atau 0 dimana menunjukkan terjadi dan tidak terjadi jatuh.

9. Klik "Start", maka *machine learning* untuk membangun *tree* akan dilakukan terhadap masukan *dataset*. Untuk lebih jelasnya dapat dilihat pada Gambar 3.22.



Gambar 3.22 Hasil Pembuatan Tree

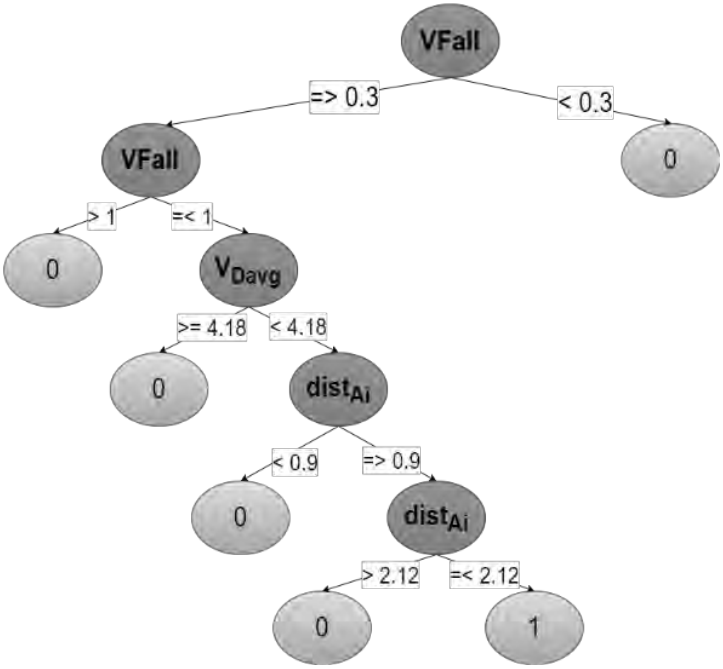
Setelah beberapa waktu, akan muncul hasil pembuatan *tree* yang dapat dilihat dengan detail pada bagian "Classifier Output" masih pada *tab* yang sama.

10. Dari hasil keluaran algoritma RandomForest, didapatkan banyak model *tree* dengan detail masing-masing dapat dievaluasi pada kotak "Summary".

Pada dasarnya, pembangunan *decision tree* bergantung pada hasil dari *dataset*. Semakin akurat *dataset* dengan pelabelan yang benar, maka pembuatan *decision tree* tidak memerlukan banyak pengubahan yang harus dilakukan.

Seperti telah dijelaskan pada langkah-langkah pembangunan *decision tree,* dari *dataset* yang telah dikumpulkan, penggunaan Weka berlanjut hingga dihasilkan keluaran hasil *tree* yang banyak menggunakan metode RandomForest. pembangunan

tree dilakukan dengan metode RandomForest dikarenakan *dataset* memiliki nilai-nilai atribut dengan angka *numeric* sehingga tidak perlu melakukan banyak *preprocess* pada *dataset* yang telah dikumpulkan. Dari banyak hasil *tree* yang dihasilkan, kemudian dipilih *tree* dengan ukuran terkecil, yang kemudian di modelkan seperti dapat dilihat pada Gambar 3.23.



Gambar 3.23 Hasil Rancangan Model Decision Tree

Dengan beberapa keterangan berikut:

- VFall = kecepatan jatuh yang terjadi, didapat dari rata-rata kecepatan dari semua titik *joint* tubuh yang telah ditentukan

- V_{Xi} = kecepatan titik X pada *frame* ke i.
- V_{Xavg} = rata-rata kecepatan titik X pada N *frame*.
- $dist_{Xi}$ = jarak titik X ke bidang normal pada *frame* ke i.
- X merupakan himpunan titik-titik yang telah ditentukan sesuai pada subbab sebelumnya, diantaranya adalah:
    a. A = titik kepala
    b. B = titik tulang belakang
    c. C = titik bahu kiri
    d. D = titik bahu kanan

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi dari analisis dan perancangan sistem yang telah dibahas pada Bab III. Namun dalam penerapannya, rancangan tersebut dapat mengalami perubahan sewaktu-waktu apabila memang dibutuhkan.

4.1. Lingkungan Implementasi

Dalam implementasinya, lingkungan yang digunakan sama seperti yang dituliskan pada rancangan, yakni menggunakan beberapa perangkat pendukung sebagai berikut.

4.1.1. Lingkungan Implementasi Perangkat Keras

Perangkat keras yang digunakan dalam implementasi pengembangan aplikasi ini adalah Komputer dan Kinect. Spesifikasi komputer yang digunakan adalah *notebook* dengan spesifikasi yang dapat dilihat pada Tabel 4.1:

Tabel 4.1 Lingkungan Implementasi Perangkat Keras

Notebook ASUS-N46VM	Prosesor	Prosesor Intel® Core™ i7-3610QM CPU @ 2.30 GHz (8 CPUs)
	Memori Primer	4096 MB
	Memori sekunder	750 GB
	Display	• NVIDIA GT 630M • Intel(R) HD Graphics 4000

4.1.2. Lingkungan Implementasi Perangkat Lunak

Penjelasan perangkat lunak yang digunakan dalam implementasi aplikasi ini adalah sebagai berikut:

1. Microsoft Windows 10 Pro 64-bit (10.0, Build 10586) sebagai sistem operasi pada *notebook*
2. Visual Studio 2015 Community sebagai IDE utama
3. MySQL untuk mengimplementasikan rancangan basis data

4. Kinect SDK versi 2.0 sebagai kakas bantu yang menghubungkan perangkat Kinect For Windows v2 dengan aplikasi pendeteksi jatuh yang dibuat dengan semua fitur yang ada di dalamnya.
5. CorelDraw X8 (64-bit).
6. Adobe Photoshop CC 2015.

4.2. Implementasi Basis Data

Subbab ini membahas tentang implementasi basis data yang telah dirancang dan dibahas pada Bab III. Basis data yang dibuat terdiri dari total sebanyak tiga tabel sesuai dengan perancangan basis data sebelumnya.

4.2.1. Implementasi Tabel Pengasuh

Pada tabel ini, sesuai dengan rancangan yang telah dibuat, tabel pengasuh memiliki beberapa atribut yang disimpan pada *database*. Detail bentuk basis data pengasuh dapat dilihat pada Gambar 4.1 dan contoh data yang telah disimpan pada *database* sesuai Gambar 4.2

	#	Name	Type	Collation	Attributes	Null	Default
☐	1	pengasuh_id	varchar(10)			No	None
☐	2	pengasuh_nama	varchar(200)			No	None
☐	3	pengasuh_email	varchar(100)			No	None
☐	4	pengasuh_password	varchar(20)			No	None
☐	5	pengasuh_tmplahir	varchar(100)			Yes	NULL
☐	6	pengasuh_tgllahir	date			Yes	NULL
☐	7	pengasuh_alamat	varchar(300)			Yes	NULL

Gambar 4.1 Tabel "Pengasuh" pada Basis Data

pengasuh_id	pengasuh_nama	pengasuh_email	pengasuh_password	pengasuh_tmplahir	pengasuh_tgllahir	pengasuh_alamat
anggosap	Angga Saputra	anggasukses@yahoo.com	bismillah	Surabaya	1994-02-05	Jalan Senayan No. 3, Surabaya
[illegible]	Christo Alan	persada@bangun.com	[illegible]	Balikpapan	1960-03-31	Jalan Kaliurang VIII 8, Yogyakarta, DIY
fahmylhor	Fahmy Thoriqul Aurima	aurimaisgold@gmail.com	fahmylhor	Madiun	1992-05-04	Jalan Mangga VII 88, Kota Madiun
[illegible]	Tresa Sunis	tresa@gmail.com	123456	Sidoarjo	1990-07-19	Jalan Pemuda

Gambar 4.2 Contoh Data Pengasuh pada Database

4.2.2. Implementasi Tabel Lansia

Pada tabel lansia, sesuai dengan rancangan yang telah dibuat, tabel lansia memiliki beberapa atribut yang disimpan pada *database*. Detail bentuk basis data lansia dapat dilihat pada Gambar 4.3 dan Gambar 4.4

#	Name	Type	Collation	Attributes	Null	Default
1	lansia_id	varchar(10)			No	*None*
2	lansia_nama	varchar(200)			No	*None*
3	lansia_tmplahir	varchar(100)			Yes	*NULL*
4	lansia_tgllahir	date			Yes	*NULL*
5	lansia_deskr	varchar(500)			Yes	*NULL*

Gambar 4.3 Tabel "Lansia" pada Basis Data

lansia_id	lansia_nama	lansia_tmplahir	lansia_tgllahir	lansia_deskr
gianto	Gianto Ginandar	Yogyakarta	1931-12-03	Sangat suka mengonsumsi makanan asin. Cukup rajin ...
jaiman	Jaiman Jalaluddin	Madiun	1941-02-07	Sangat trauma terhadap beberapa hal karena memilik...
joyo	Joyo Subianto	Kediri	1937-05-12	Memiliki gangguan pada persendian sehingga kesusah...
jumanto	Jumanto Sumanto	Madiun	1933-11-01	Sangat aktif dalam berolahraga namun memiliki pem...
muliono	Muliono Sunu R.	Ngawi	1943-01-01	memiliki berat badan berlebih/ obesitas. Sangat ke...

Gambar 4.4 Contoh Data Lansia dalam Database

4.2.3. Implementasi Tabel Jatuh

Sesuai dengan perancangan sebelumnya karena hasil relasi *may to many* antara tabel lansia dengan tabel pengasuh maka terbentuklah tabel baru yaitu tabel ajaruh. Pada tabel ini, sesuai dengan rancangan yang telah dibuat, tabel jatuh memiliki beberapa

atribut yang disimpan pada *database*. Detail bentuk basis data jatuh dapat dilihat pada Gambar 4.3 dan Gambar 4.4

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	jatuh_id	int(11)			No	None	AUTO_INCREMENT
2	pengasuh_id	varchar(10)			No	None	
3	lansia_id	varchar(10)			No	None	
4	jatuh_waktu	datetime			No	None	

Gambar 4.5 Tabel "Jatuh" pada Basis Data

jatuh_id	pengasuh_id	lansia_id	jatuh_waktu
1	fahmythor	jaiman	2016-06-06 00:00:00
2	anggosap	jumanto	2016-06-07 01:40:09
3	fahmythor	muliono	2016-06-07 16:13:07
4	anggosap	muliono	2012-05-15 08:06:26

Gambar 4.6 Contoh Data Jatuh dalam Database

4.3. Implementasi Proses

Subab ini akan menjelaskan mengenai implementasi proses yang terjadi dalam aplikasi sesuai rancangan yang telah dibahas sebelumnya pada Bab III.

4.3.1. Implementasi Mendaftar Pada Sistem

Mendaftar pada sistem adalah yang dilakukan pengguna/ pengasuh ketika pertama kali menggunakan sistem. Mendaftar diawali dengan keluarnya jendela registrasi yang meminta pengasuh untuk mengisi masukan-masukan berisi data-data untuk membuat akun pada sistem. Setelah memasukkan pada kotak masukan yang disediakan dan melanjutkan proses mendaftar, maka sistem akan mengeksekusi *query* ke *database* untuk mendaftarkan

akun sesuai masukan. Implementasi ini dapa dilihat pada Kode Sumber 4.1 di bawah.

```
1  private void btn_registrasi_Click(object sender,
   RoutedEventArgs e)
2  {
3      try
4      {
5          string namapengasuh = txtbox_NamaPengasuh_reg.Text;
6          string idpengasuh = txtbox_IDPengasuh_reg.Text;
7          string tmplahirpengasuh =
   txtbox_TmpLahirPengasuh_reg.Text;
8          string tgllahirpengasuh =
   datepick_TglLahirPengasuh_reg.SelectedDate.Value.Date.ToShortD
   ateString();
9          string passwordpengasuh =
   txtbox_PasswordPengasuh_reg.Password;
10         string emailpengasuh = txtbox_EmailPengasuh_reg.Text;
11         string alamatpengasuh =
   txtbox_AlamatPengasuh_reg.Text;
12
13         OpenConn();
14         string sql = "INSERT INTO pengasuh (pengasuh_id,
   pengasuh_nama, pengasuh_email, pengasuh_password,
   pengasuh_tmplahir, pengasuh_tgllahir, pengasuh_alamat) VALUES
   ('" + idpengasuh + "', '" + namapengasuh + "', '" +
   emailpengasuh + "','" + passwordpengasuh + "','" +
   tmplahirpengasuh + "', STR_TO_DATE('" + tgllahirpengasuh + "',
   '%d/%m/%Y'), '" + alamatpengasuh + "');";
15         MySqlCommand jalan = new MySqlCommand(sql, koneksi);
16         jalan.ExecuteNonQuery();
17
18         statusdaftar = true;
19     }
20     catch (Exception ex)
21     {
22         this.ShowMessageAsync("Terjadi Kesalahan",
23          ex.Message,
24          MessageDialogStyle.Affirmative, msgBoxSetting);
25     }
27     this.Close();
29 }
```

Kode Sumber 4.1 Implementasi Daftar ke Sistem

4.3.2. Implementasi Masuk Dalam Sistem

Masuk pada sistem adalah ketika pengasuh memasukkan identifikasi *username* dan *password* diikuti menekan tombol *login*. Setelah itu, sistem akan melakukan *query* pengecekan apakah akun dengan identifikasi sesuai masukan ada dalam *database*. Implementasi dalam bentuk kode pada proses ini dapat dilihat pada Kode Sumber 4.2 dan Kode Sumber 4.3.

```
1  private void btn_masuk_Click(object sender, RoutedEventArgs e)
2  {
3      OpenConn();
5      try
6      {
7          if (statuskoneksi)
8          {
9              if (!txtbox_Password.Password.Equals("") &&
   !txtbox_Username.Text.Equals(""))
10             {
11                 string cekLogin = "SELECT pengasuh_id FROM
   pengasuh where pengasuh_id='" + txtbox_Username.Text + "' and
   pengasuh_password='" + txtbox_Password.Password + "'";
12                 MySqlCommand jalan = new
   MySqlCommand(cekLogin, koneksi);
13                 MySqlDataReader rdr = jalan.ExecuteReader();
14
15                 if (rdr.HasRows)
16                 {
17                     string cekX = null;
18                     if (rdr.Read())
19                         cekX = rdr[0].ToString();
21                     koneksi.Close();
23                     Bismillah_Main test = new
   Bismillah_Main(cekX);
24                     test.Show();
25                     this.Close();
27                 }
28                 else {
29                     this.ShowMessageAsync("An Error Occured",
31                      "Username and Password Combination is not
   Registered or false",
```

Kode Sumber 4.2 Implementasi Login ke Sistem (a)

```
32                      MessageDialogStyle.Affirmative,
    msgBoxSetting);
33                  }
35              }
36              else {
37                  var msgBoxSetting = new MetroDialogSettings()
38                  {
39                      AffirmativeButtonText = "OK",
40                      AnimateShow = true,
41                      AnimateHide = true,
42                  };
43                  this.ShowMessageAsync("An Error Occured",
45                   "Sorry username and password isn't match and
    or it hasn't registered!",
47                   MessageDialogStyle.Affirmative,
    msgBoxSetting);
48              }
49          }
50      }
51      catch (InvalidOperationException ex)
53      {
55          this.ShowMessageAsync("An error occured",
56           ex.Message,
57           MessageDialogStyle.Affirmative, msgBoxSetting);
58      }
59  }
```

Kode Sumber 4.3 Implementasi Login ke Sistem (b)

4.3.3. Implementasi Mengaktifkan Kinect

Pada jendela utama, ketika pengasuh menekan tombol pemicu aktifkan akan mengaktifkan mode pengawasan dan Kinect akan mulai menangkap gambar. Diawali dengan sistem yang mencari sumber Kinect seperti bisa dilihat pada Kode Sumber 4.4 dan Kode Sumber 4.5.

```
1  tes = KinectSensor.GetDefault();
2  tes.Open();
3  bisaNyala = false;
4  colorReader = tes.ColorFrameSource.OpenReader();
```

Kode Sumber 4.4 Implementasi Aktifkan Kinect (a)

```
5   fdes =
    tes.ColorFrameSource.CreateFrameDescription(ColorImageFormat.B
    gra);
6
7   colorBitmap = new WriteableBitmap(fdes.Width, fdes.Height,
    96.0, 96.0, PixelFormats.Bgr32, null);
8   image.Source = colorBitmap;
9
10  this.titikBadan = new List<string>();
```

Kode Sumber 4.5 Implementasi Aktifkan Kinect (b)

4.3.4. Implementasi Proses Mendeteksi Jatuh

Impelementasi dari proses mendeteksi jatuh merupakan pengabungan dari implementasi 4.3.5 sampai 4.3.9. Mendeteksi jatuh juga merupakan implementasi algoritma pendeteksian jatuh yang telah dijelaskan sebelumnya pada bab III. Proses ini pada dasarnya adalah suatu kesatuan dari berbagai proses yang akan dijelaskan berikutnya.

4.3.5. Implementasi Proses Tracking Badan

Implementasi *tracking* badan dilakukan ketika Kinect telah berhasil menangkap *frame* tubuh yang akan dideteksi. Setelah menangkap *frame* tubuh maka aplikasi akan menampilkan tangkapan Kinect tersebut pada sebuah gambar bergerak *realtime* sesuai tangkapan Kinect. Kemudian aplikasi akan menandai tubuh yang berhasil ditangkap dengan titik-titik pada tubuh. Implementasi ini dapat dilihat pada Kode Sumber 4.6, Kode Sumber 4.7, Kode Sumber 4.8, dan Kode Sumber 4.9.

```
1  bool dataReceived = false;
2  List<float> dummy = new List<float>();
3  MultiSourceFrame msf = e.FrameReference.AcquireFrame();
4  if (msf != null) {
5   using(BodyFrame bodyFrame =
   msf.BodyFrameReference.AcquireFrame()) {
6     using(ColorFrame colorFrame =
   msf.ColorFrameReference.AcquireFrame()) {
7      if (colorFrame != null && bodyFrame != null) {
8       dummy.Add(bodyFrame.FloorClipPlane.X);
```

Kode Sumber 4.6 Implementasi Tracking Tubuh (a)

```
9       dummy.Add(bodyFrame.FloorClipPlane.Y);
10      dummy.Add(bodyFrame.FloorClipPlane.Z);
11      dummy.Add(bodyFrame.FloorClipPlane.W);
12      this.bodies = new Body[bodyFrame.BodyCount];
13      FrameDescription colorFrameDescription =
    colorFrame.FrameDescription;
14      using(KinectBuffer colorBuffer =
    colorFrame.LockRawImageBuffer()) {
15       this.colorBitmap.Lock();
16       if ((colorFrame.FrameDescription.Width ==
    this.colorBitmap.PixelWidth)) {
17     colorFrame.CopyConvertedFrameDataToIntPtr(
18      this.colorBitmap.BackBuffer,
19      (uint)(colorFrame.FrameDescription.Width *
    colorFrame.FrameDescription.Height * 4),
20      ColorImageFormat.Bgra);
21     this.colorBitmap.AddDirtyRect(new Int32Rect(0, 0,
    this.colorBitmap.PixelWidth, this.colorBitmap.PixelHeight));
22  }
23       this.colorBitmap.Unlock();
24  }
25  bodyFrame.GetAndRefreshBodyData(this.bodies);
26  bodyCanvas.Children.Clear();
27  dataReceived = true;
28  xcurrentts = new TimeSpan(bodyFrame.RelativeTime.Ticks
    currentTs = xcurrentts.Milliseconds; deltaTs = (currentTs -
    previousTs) / 1000;
29      }
30      if (currentFrame >= totalFrame) {
31       for (int i = 0; i<jointTubuhArr.Count; i++) {
32        jointTubuhArr[i].velocityAVG = cariKecRtJoint(i);
33  jointTubuhArr[i].deltaResultVelocityAVG.Clear();
34       }
35       currentFrame = 0;
36      }
37     }
38    }
39    if (dataReceived) {
40     Body body = null;
41     if (this.bodyTracked) {
42     if (this.bodies[this.bodyIndex].IsTracked)
```

Kode Sumber 4.7 Implementasi Tracking Tubuh (b)

```
    {
        body = this.bodies[this.bodyIndex];
    }
    else {
        bodyTracked = false;
    }
}
    if (!bodyTracked) {
    for (int i = 0; i < this.bodies.Length; ++i)
    {
        if (this.bodies[i].IsTracked)
        {
            this.bodyIndex = i;
            this.bodyTracked = true;
            break;
        }
    }
}
    if (body != null && this.bodyTracked && body.IsTracked) {
    if (body.IsTracked)
    {
        float temp =
         id = 0;
        foreach (Joint joint in body.Joints.Values)
        {
            if (joint.TrackingState == TrackingState.Tracked ||
joint.TrackingState == TrackingState.Inferred)
            {
                CameraSpacePoint jointPosition = new
CameraSpacePoint();
                jointPosition = joint.Position;
                Point point = new Point();
                ColorSpacePoint colorPoint =
tes.CoordinateMapper.MapCameraPointToColorSpace(jointPosition);
                point.X = float.IsInfinity(colorPoint.X) ? 0 :
colorPoint.X;
                point.Y = float.IsInfinity(colorPoint.Y) ? 0 :
colorPoint.Y;
bool dataReceived = false;
List<float> dummy = new List<float>();
```

Kode Sumber 4.8 Implementasi Tracking Tubuh (c)

```
78 MultiSourceFrame msf = e.FrameReference.AcquireFrame();
79 if (msf != null) {
80  using(BodyFrame bodyFrame =
   msf.BodyFrameReference.AcquireFrame()) {
81    using(ColorFrame colorFrame =
   msf.ColorFrameReference.AcquireFrame()) {
82     if (colorFrame != null && bodyFrame != null) {
83      dummy.Add(bodyFrame.FloorClipPlane.X);
84      dummy.Add(bodyFrame.FloorClipPlane.Y);
85      dummy.Add(bodyFrame.FloorClipPlane.Z);
86      dummy.Add(bodyFrame.FloorClipPlane.W);
87      this.bodies = new Body[bodyFrame.BodyCount];
88      FrameDescription colorFrameDescription =
   colorFrame.FrameDescription;
89      using(KinectBuffer colorBuffer =
   colorFrame.LockRawImageBuffer()) {
90       this.colorBitmap.Lock();
```

Kode Sumber 4.9 Implementasi Tracking Tubuh (d)

4.3.6. Implementasi Penentuan Titik Titik Tubuh Utama

Penentuan titik-titik pada tubuh dilakukan setelah mendapatkan tubuh yang *tracked*. Pada proses ini ditentukanlah titik-titik yang memiliki *gain information* tertinggi dibandingkan titik *joint* tubuh lain. Titik ini antara lain adalah titik *joint* kepala, bahu kanan, bahu kiri, dan tulang belakang. Implementasi dapat dilihat pada Kode Sumber 4.10 di bawah.

```
1  if (joint.JointType == JointType.Head || joint.JointType ==
   JointType.SpineShoulder || joint.JointType ==
   JointType.ShoulderLeft || joint.JointType ==
   JointType.ShoulderRight)
```

Kode Sumber 4.10 Implementasi Penentuan Titik Joint

4.3.7. Implementasi Proses Mencari Jarak Titik Terhadap Bidang Normal

Proses mencari jarak titik terhadap bidang normal dilakukan pada langkah setelah menentukan titik-titik *joint* tubuh. Imeplementasi proses ini dapat dilihat pada Kode Sumber 4.11.

```
1  private float cariJarak(List<float> dummy, Joint joint)
2  {
3      float A = dummy[0];
4      float B = dummy[1];
5      float C = dummy[2];
6      float D = dummy[3];
7      float x = (float)((A * joint.Position.X) + (B *
   joint.Position.Y) + (C * joint.Position.Y) + D) /
   (float)(Math.Sqrt((Math.Pow(A, 2) + Math.Pow(B, 2) +
   Math.Pow(C, 2))));
8      return x;
9  }
```

Kode Sumber 4.11 Implementasi Cari Jarak Joint

4.3.8. Implementasi Menentukan Kecepatan Rata-Rata Titik Joint Tubuh

Setelah mendapatkan jarak titik dengan bidang normal pada *frame* ke-i dan ke-i+1, akan ditemukan selisih jarak yang kemudian ditentukan kecepatan tiap *joint* terhadap waktu relatif. Kemudian proses ini berlanjut pada menentukan kecepatan rata-rata semua *joint* yang telah ditentukan. Implementasi dapat dilihat pada Kode Sumber 4.12.

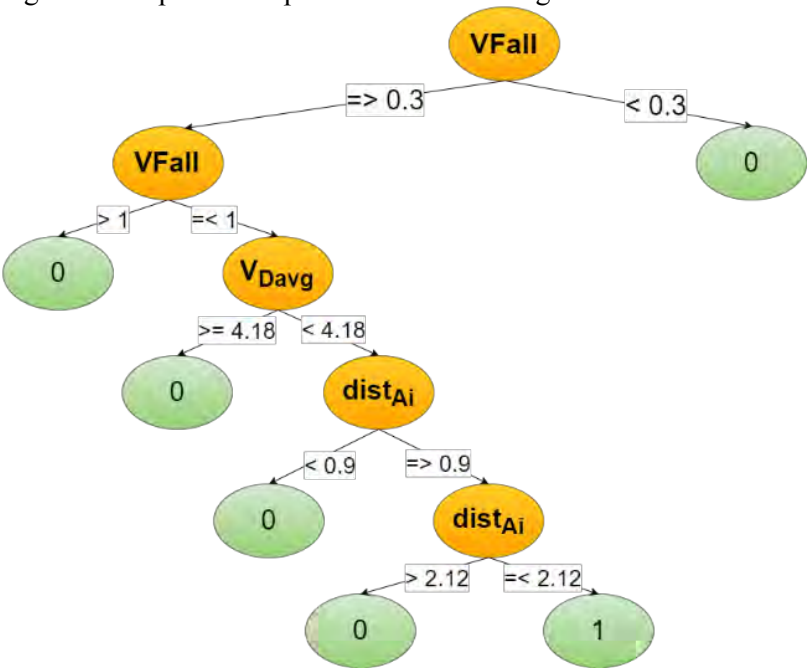
```
1  public float cariKecRtJoint(int id)
2  {
3      float total = 0;
4      float x = 0;
5      if (jointTubuhArr.Count >= id)
6      {
7          for (int i = 0; i < totalFrame; i++)
8          {
9              if (jointTubuhArr[id].deltaResultVelocityAVG.Count
   >= totalFrame)
10             {
11               total+ =
   jointTubuhArr[id].deltaResultVelocityAVG[i];
14             }
15         }
16         x = total / totalFrame;
17     }
18     return x; }
```

Kode Sumber 4.12 Implementasi Cari Kecepatan Rata-rata

4.3.9. Implementasi Decision Tree Kejadian Jatuh

Proses ini adalah proses terakhir yang akan dilakukan dalam penentuan kejadian jatuh. Pada proses ini, data-data yang diperoleh dari subproses-subproses Proses mendeteksi jatuh pada *decision tree* yang akhirnya akan menentukan apakah telah terjadi jatuh atau tidak.

Impelementasi *decision tree* kejadian jatuh adalah hasil dari implementasi pohon keputusan untuk mendapatkan keputusan terjadi jatuh atau tidak setelah didapatkan atribut-atribut nilai variabel yang dibutuhkan untuk *decision tree* ini. Pada implementasinya, pohon keputusan ini merupakan hasil yang mengacu pada rancangan di Bab III. Model dari *decision tree* yang digunakan dapat dilihat pada Gambar 4.7 dengan



Gambar 4.7 Decision Tree Yang diimplementasikan Pada Aplikasi

Kemudian Implementasi *decision tree* dalam bentuk kode dapat dilihat pada Kode Sumber 4.13 sebagai berikut.

```
1  private bool decisionTree_Bismillah(List<JointTubuh> joinTBH)
2          {
3              bool x = false;
4              float kecJatuh = cariKecJatuh(joinTBH);
5              if (kecJatuh < 0.3)
6              {
7                  x = false;
8              }
9              else {
10                 if (kecJatuh > 1)
11                     x = false;
14                 else {
15                     if (joinTBH[3].velocityAVG < 4.18)
16                     {
17                         if (joinTBH[0].currDist < 0.9)
18                             x = false;
19                         else {
20                             if (joinTBH[0].currDist > 2.05)
21                                 x = false;
22                             else
23                                 x = true;
24                         }
25                     }
26                     else
27                         x = false;
28                 }
29             }
30             return x;
31         }
```

Kode Sumber 4.13 Decision Tree Keputusan Jatuh

4.3.10. Implementasi Mengelola Data Lansia

Implementasi mengelola data lansia meliputi implementasi pada proses 4.3.11

4.3.11. Implementasi Menambah, atau Mengupdate, atau Menghapus

Pada proses ini pengasuh bisa melakukan penambahan, pengubahan hingga penghapusan data lansia. Dengan mengisikan data lansia baru, pengasuh bisa menambah data. Kemudian dengan memilih data yang akan dihapus atau diubah maka sistem akan menghapus atau mengubah dan menyimpan di *store* lansia. Implementasi proses ini dapat dilihat pada Kode Sumber 4.14 dan Kode Sumber 4.15 untuk proses mengubah, Kode Sumber 4.16 dan Kode Sumber 4.17 untuk hapus, dan Kode Sumber 4.18 dan Kode Sumber 4.19 untuk menambah data lansia baru.

```
1  private void btn_update_Click(object sender, RoutedEventArgs e)
   {
2   string namalansia = null;
3   string idlansia = null;
4   string tmplahirlansia = null;
5   string tgllahirlansia = null;
6   string deskripsilansia = null;
7   try {
8    openConn();
9
10   namalansia = txtbox_NamaLansia.Text;
11   idlansia = txtbox_IDLansia.Text;
14   tmplahirlansia = txtbox_TmpLahir.Text;
15   tgllahirlansia =
   datepick_TglLahirLansia.SelectedDate.Value.Date.ToShortDateStri
   ng();
16   deskripsilansia = new
   TextRange(richTextBox_deskripsi.Document.ContentStart,
   richTextBox_deskripsi.Document.ContentEnd).Text;
17
18   string sql = "UPDATE lansia SET lansia_id = '" + idlansia +
   "', lansia_nama = '" + namalansia + "', lansia_tmplahir = '" +
   tmplahirlansia + "', lansia_tgllahir = STR_TO_DATE('" +
   tgllahirlansia + "', '%d/%m/%Y'), lansia_deskr = '" +
   deskripsilansia + "' WHERE lansia_id = '" + idlansia + "';";
19   MySqlCommand jalan = new MySqlCommand(sql, koneksi);
20   jalan.ExecuteNonQuery();
21   koneksi.Close();
```

Kode Sumber 4.14 Impelementasi Update Data Lansia (a)

```
22
23   isiDtLansia();
24   updatelansia = true;
25  } catch (Exception ex) {
26   this.ShowMessageAsync("Terjadi Kesalahan",
27    ex.Message,
28    MessageDialogStyle.Affirmative, msgBoxSetting);
29  } finally {
30   if (updatelansia) {
31    this.ShowMessageAsync("Success",
32     "Elder " + idlansia + "'s Profile has been updated!",
33     MessageDialogStyle.Affirmative, msgBoxSetting);
34    updatelansia = false;
35   }
36
37   datagrid_Lansia.SelectedIndex = datalansiaselected_combobox;
38  }
39
40 }
41 private void btn_update_Click(object sender, RoutedEventArgs
   e) {
42  string namalansia = null;
43  string idlansia = null;
44  string tmplahirlansia = null;
45  string tgllahirlansia = null;
46  string deskripsilansia = null;
47  try {
```

Kode Sumber 4.15 Impelementasi Update Data Lansia (b)

```
1  private void btn_delete_Click(object sender, RoutedEventArgs e)
2  {
3      string idlansia = null;
4      try
5      {
6          openConn();
7          idlansia = txtbox_IDLansia.Text;
8          string sql = "DELETE FROM lansia where lansia_id = '" +
   idlansia + "';";
9          MySqlCommand jalan = new MySqlCommand(sql, koneksi);
```

Kode Sumber 4.16 Implementasi Hapus Data Lansia (a)

```
10          jalan.ExecuteNonQuery();
11          koneksi.Close();
14
15          isiDtLansia();
16          hapuslansia = true;
17      }
18      catch (Exception ex)
19      {
20          this.ShowMessageAsync("Terjadi Kesalahan",
21           ex.Message,
22           MessageDialogStyle.Affirmative, msgBoxSetting);
23      }
24      finally
25      {
26          if (hapuslansia)
27          {
28              this.ShowMessageAsync("Success",
29               "Elder " + idlansia + "'s has been deleted from
    database!",
30               MessageDialogStyle.Affirmative, msgBoxSetting);
31              hapuslansia = false;
32
33              txtbox_IDLansia.Clear();
34              txtbox_IDLansia.IsReadOnly = true;
35
36              txtbox_NamaLansia.Clear();
37              txtbox_TmpLahir.Clear();
38              datepick_TglLahirLansia.SelectedDate = null;
39              richTextBox_deskripsi.Document.Blocks.Clear();
40          }
41      }
```

Kode Sumber 4.17 Implementasi Hapus Data Lansia (b)

```
1  private void btn_tambahlansia_Click(object sender,
   RoutedEventArgs e)
2  {
3      try
4      {
5          string namalansia = txtbox_NamaLansia_tmbh.Text;
```

Kode Sumber 4.18 Implementasi Tambah Lansia (a)

```
6           string idlansia = txtbox_IDLansia_tmbh.Text;
7           string tmplahirlansia =
txtbox_TmpLahirLansia_tmbh.Text;
8           string tgllahirlansia =
datepick_TglLahirLansia_reg.SelectedDate.Value.Date.ToShortD
ateString();
9           string deskripsilansia = new
TextRange(richTextBox_deskripsi_tmbh.Document.ContentStart,
richTextBox_deskripsi_tmbh.Document.ContentEnd).Text;
10
11          OpenConn();
14
15          string sql = "INSERT INTO lansia (lansia_id,
lansia_nama, lansia_tmplahir, lansia_tgllahir, lansia_deskr)
VALUES ('" + idlansia + "', '" + namalansia + "', '" +
tmplahirlansia + "', STR_TO_DATE('" + tgllahirlansia + "',
'%d/%m/%Y'), '" + deskripsilansia + "');";
16          MySqlCommand jalan = new MySqlCommand(sql, koneksi);
17          jalan.ExecuteNonQuery();
18          tambahlansia = true;
19      }
20      catch (Exception ex)
21      {
22          this.ShowMessageAsync("Terjadi Kesalahan",
23           ex.Message,
24           MessageDialogStyle.Affirmative, msgBoxSetting);
25      }
26
27      this.Close();
28  }
```

Kode Sumber 4.19 Implementasi Tambah Lansia (b)

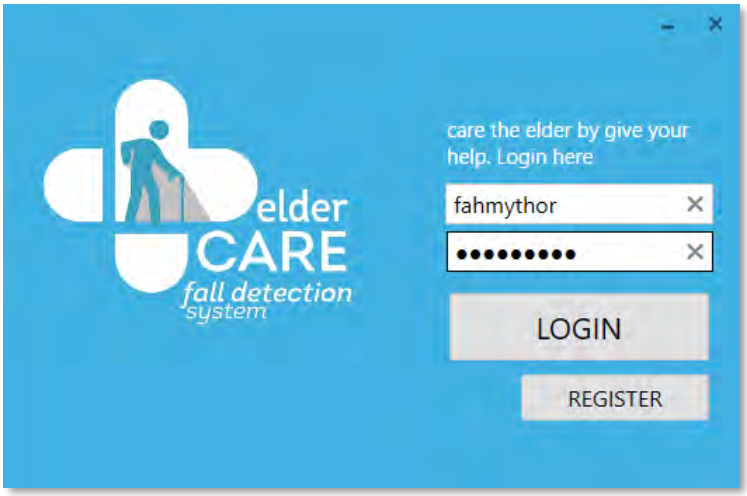
4.4. Implementasi Tampilan Antarmuka

Subbab ini membahas tentang implementasi tampilan antarmuka yang telah dirancang dan dibahas pada Bab III. Selanjutnya akan dirinci berdasarkan urutan halaman yang akan tampil dan dilihat oleh pengguna/ pengasuh.

4.4.1. Implementasi Jendela Login

Pada jendela ini pengasuh diminta untuk memasukkan masukan *username* dan *password* untuk dapat masuk ke dalam

aplikasi. Jendela ini menampilkan logo aplikasi, kotak masukan *username,* kotak masukan *password*, tombol *login*, dan tombol *register* Seperti bisa dilihat pada Gambar 4.8. Kode sumber jendela *login* dapat dilihat pada Kode Sumber 4.20 dan Kode Sumber 4.21.



Gambar 4.8 Jendela Login

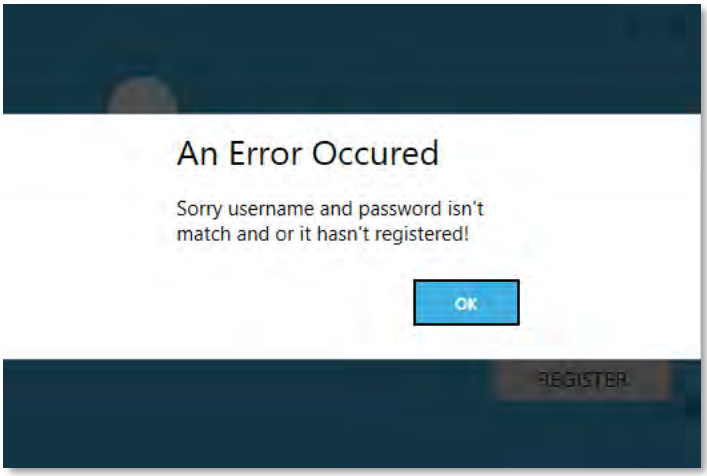
```
1  <Grid Background="#FF41B1E1">
2                      <Grid.RowDefinitions>
3                          <RowDefinition Height="Auto" />
4                          <RowDefinition Height="*" />
5                          <RowDefinition Height="Auto" />
6          </Grid.RowDefinitions>
7          <Button x:Name="btn_masuk" Content="Login"
   Margin="315,207,19,0" Grid.Row="1" VerticalAlignment="Top"
   Width="185" FontSize="21.333" Height="48"
   Click="btn_masuk_Click" TabIndex="3" FontWeight="Normal"
   Background="#FFE0E0E0"/>
8          <Button x:Name="btn_daftar" Content="register"
   Margin="366,265,19,0" Grid.Row="1" VerticalAlignment="Top"
   Width="134" FontSize="16" Height="33" TabIndex="4"
   Click="btn_daftar_Click" FontWeight="Normal"
   Background="#FFE0E0E0"/>
```

Kode Sumber 4.20 Menampilkan Jendela Login (a)

```
9           <TextBlock x:Name="textBlock"
    HorizontalAlignment="Left" Margin="316,80,0,0" Grid.Row="1"
    TextWrapping="Wrap" Text="care the elder by give your help.
    Login here" VerticalAlignment="Top" Height="44" Width="184"
    FontSize="14.667" FontFamily="Segoe UI Symbol"
    Foreground="White" LineHeight="13.333"/>
10
11          <TextBox x:Name="txtbox_Username"
    Controls:TextBoxHelper.Watermark="Username"
    Controls:TextBoxHelper.SelectAllOnFocus="True"
    Controls:TextBoxHelper.ClearTextButton="True"
    Margin="315,129,19,0" Grid.Row="1" FontSize="16"
    TabIndex="1" VerticalAlignment="Top" />
12          <PasswordBox x:Name="txtbox_Password"
    Controls:TextBoxHelper.Watermark="Password"
    Controls:TextBoxHelper.SelectAllOnFocus="True"
    Controls:TextBoxHelper.ClearTextButton="True"
    Margin="316,163,19,0" Grid.Row="1" FontSize="16"
    TabIndex="2" VerticalAlignment="Top"/>
13                              <Image x:Name="image"
    HorizontalAlignment="Left" Height="239" Margin="0,32,0,0"
    Grid.RowSpan="2" VerticalAlignment="Top" Width="277"
    Source="Untitled-12.jpg">
14                          <Image.OpacityMask>
15                          <LinearGradientBrush EndPoint="0.5,1"
    StartPoint="0.5,0">
16                              <GradientStop Color="Black"
    Offset="0"/>
17                              <GradientStop Color="White"
    Offset="1"/>
18                          </LinearGradientBrush>
19                          </Image.OpacityMask>
20          </Image>
21 </Grid>
```

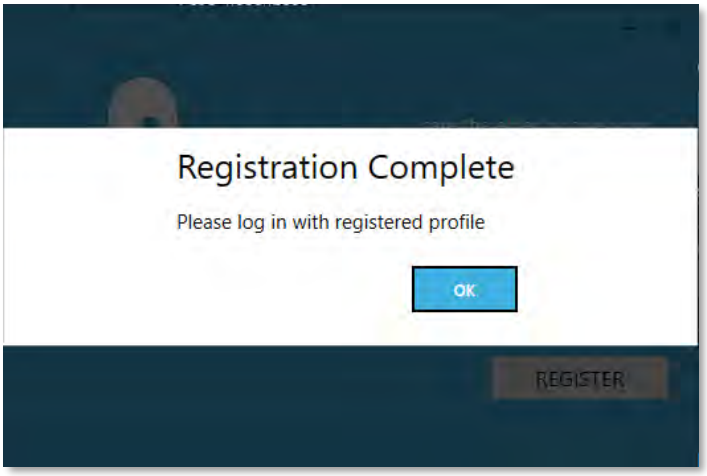
Kode Sumber 4.21 Menampilkan Jendela Login (a)

Tombol *login* akan memroses isi masukan pada kotak masukan *username* dan *password* untuk menentukan apakah kombinasi *username* sudah sesuai dengan yang telah tersimpan pada *database* atau belum. Jika sesuai, maka tombol ini akan membawa ke jendela utama. Sebaliknya, apabila belum terdaftar maupun ada kesalahan kombinasi *username* dengan *password*, akan muncul *popup* adanya kesalahan pada kombinasi, sesuai pada Gambar 4.9.



Gambar 4.9 Popup Kesalahan Login

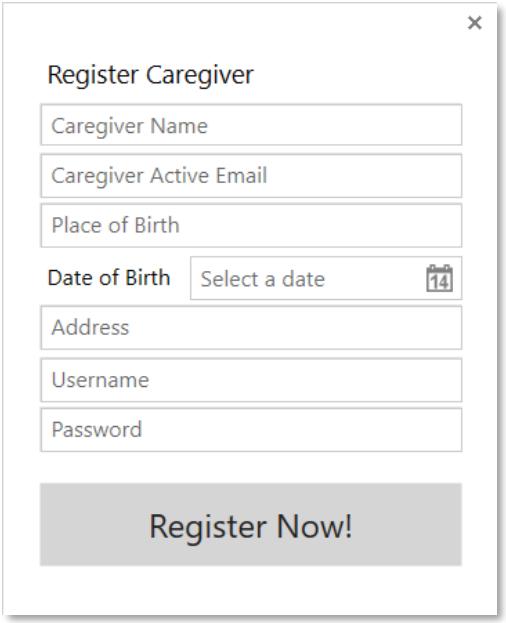
Sedangkan tombol *register* akan memanggil jendela registrasi. Setelah berhasil registrasi, akan muncul *popup* untuk *login* menggunakan akun terdaftar seperti ditunjukkan pada Gambar 4.10.



Gambar 4.10 Popup Berhasil Registrasi

4.4.2. Implementasi Jendela Registrasi

Jendela registrasi adalah jendela yang musul akibat pemicu tombol "register" pada jendela *login* sebelumnya. Pada jendela ini, pengasuh diminta untuk mendaftarkan profil diri dengan mengisi kotak masukan mulai dari nama, hingga menentukan *username* dan *password* yang nantinya digunakan untuk masuk mengakses aplikasi. Tombol "Register Now!" ditekan setelah semua kotak masukan terisi dengan benar. Untuk lebih detail dari antarmuka ini dapat dilihat pada Gambar 4.11.



Gambar 4.11 Jendela Registrasi

Sedangkan untuk kode sumber jendela registrasi bisa dilihat pada Kode Sumber 4.22, Kode Sumber 4.23, hingga Kode Sumber 4.24.

```
1 <Grid Margin="0,0,-6,2" >
2         <Grid.Resources>
3             <ResourceDictionary>
```

Kode Sumber 4.22 Menampilkan Jendela Registrasi (a)

```
4                 <ResourceDictionary.MergedDictionaries>

5                     <ResourceDictionary
Source="pack://application:,,,/MahApps.Metro;component/Styles/
FlatButton.xaml" />
6                 </ResourceDictionary.MergedDictionaries>

7             </ResourceDictionary>

8         </Grid.Resources>

9         <TextBox x:Name="txtbox_NamaPengasuh_reg"
Controls:TextBoxHelper.Watermark="Caregiver Name"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="24,59,29,326" FontSize="16" TabIndex="2"
HorizontalAlignment="Center" VerticalAlignment="Center"
Width="305" Height="30" />
10        <TextBox x:Name="txtbox_EmailPengasuh_reg"
Controls:TextBoxHelper.Watermark="Caregiver Active Email"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="24,95,29,288" FontSize="16" TabIndex="2"
HorizontalAlignment="Center" VerticalAlignment="Center"
Width="305" Height="32" />
11        <TextBox x:Name="txtbox_TmpLahirPengasuh_reg"
Controls:TextBoxHelper.Watermark="Place of Birth"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="24,131,29,252" FontSize="16" TabIndex="2"
HorizontalAlignment="Center" VerticalAlignment="Center"
Width="305" Height="32" />
12        <DatePicker x:Name="datepick_TglLahirPengasuh_reg"
HorizontalAlignment="Center" Margin="132,169,29,214"
VerticalAlignment="Center" Width="197" Height="32"
FontSize="16"/>
13        <TextBox x:Name="txtbox_IDPengasuh_reg"
Controls:TextBoxHelper.Watermark="Username"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="24,243,29,140" FontSize="16" TabIndex="1"
HorizontalAlignment="Center" VerticalAlignment="Center"
Width="305" Height="32" />
14        <PasswordBox x:Name="txtbox_PasswordPengasuh_reg"
Controls:TextBoxHelper.Watermark="Password"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="24,279,29,104" FontSize="16" TabIndex="1"
HorizontalAlignment="Center" VerticalAlignment="Center"
Width="305" Height="32" />
```

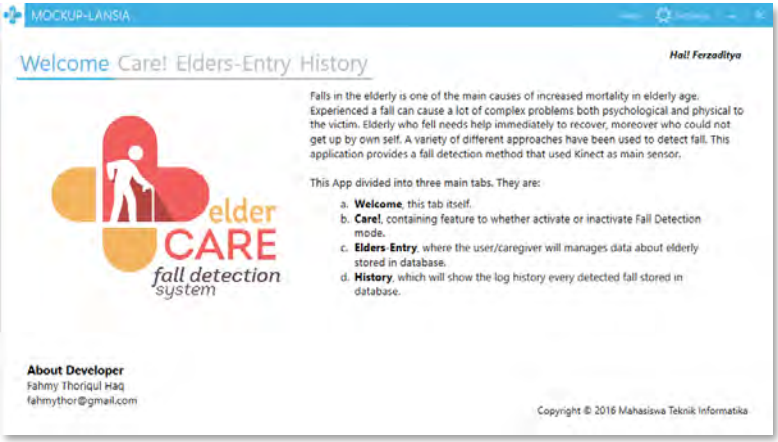
Kode Sumber 4.23 Menampilkan Jendela Registrasi (b)

```
15          <TextBox x:Name="txtbox_AlamatPengasuh_reg"
    Controls:TextBoxHelper.Watermark="Address"
    Controls:TextBoxHelper.SelectAllOnFocus="True"
    Margin="24,205,29,178" FontSize="16" TabIndex="1"
    HorizontalAlignment="Center" VerticalAlignment="Center"
    Width="305" Height="32" />
16          <Button x:Name="btn_registrasi" Content="Register
    Now!" HorizontalAlignment="Center" Margin="24,334,29,21"
    VerticalAlignment="Center" Width="305" FontSize="24"
    Height="60" TabIndex="3" Click="btn_registrasi_Click"/>
17          <Label x:Name="label" Content="Register Caregiver"
    HorizontalAlignment="Center" Margin="24,19,75,360"
    VerticalAlignment="Center" Height="36" FontSize="18.667"
    Width="259"/>
18          <Label x:Name="label1" Content="Date of Birth"
    HorizontalAlignment="Center" Margin="24,169,231,217"
    VerticalAlignment="Center" FontSize="16" Width="103"/>
19
20      </Grid>
```

Kode Sumber 4.24 Menampilkan Jendela Registrasi (c)

4.4.3. Implementasi Jendela Utama – Tab Tentang

Jendela utama terdiri dari empat *tab* sesuai dengan pada bab perancangan sebelumnya. Pada *tab* "Tentang" terjadi perubahan nama menjadi *tab* "Welcome".



Gambar 4.12 Jendela Utama – Tab Tentang

Sesuai pada Gambar 4.12, Konten utama pada *tab* ini adalah deskripsi singkat mengenai aplikasi pendeteksi jatuh. Kemudian implementasi detail seperti ditunjukkan pada Kode Sumber 4.25 dan Kode Sumber 4.26.

```
1  Controls:MetroTabItem Header="Welcome">
2                      <Grid>
3                          <Grid.ColumnDefinitions>
4                              <ColumnDefinition/>
5                          </Grid.ColumnDefinitions>
6                          <TextBlock TextWrapping="Wrap"
   TextAlignment="Left" FontSize="13.333" Margin="355,6,10,270"
   ><Run Text="Falls in the elderly is one of the main causes of
   increased mortality in elderly age. Experienced a fall can
   cause a lot of complex problems both psychological and
   physical to the victim. Elderly who fell needs help
   immediately to recover, moreover who could not get up by own
   self. A variety of different approaches have been used to
   detect fall. This application provides a fall detection method
   that used Kinect as main
   sensor."/><LineBreak/><Run/><LineBreak/><Run Text="This App
   divided into three main tabs. They are:"/></TextBlock>
7                          <Image x:Name="image1"
   HorizontalAlignment="Left" Height="279" Margin="0,11,0,0"
   VerticalAlignment="Top" Width="355" Source="Untitled-
   123.jpg"/>
8                          <TextBlock x:Name="textBlock"
   HorizontalAlignment="Left" Margin="409,138,0,0"
   TextWrapping="Wrap" VerticalAlignment="Top" Width="423"
   Height="147" FontSize="13.333"><Run FontWeight="Bold"
   Text="Welcome"/><Run Text=", this tab itself. &#xA;"/><Run
   FontWeight="Bold" Text="Care!"/><Run Text=", containing
   feature to whether activate or inactivate Fall Detection
   mode.&#xA;"/><Run FontWeight="Bold" Text="Elders-Entry"/><Run
   Text=", where the user/caregiver will manages data about
   elderly stored in database.&#xA;"/><Run FontWeight="Bold"
   Text="History"/><Run Text=", which will show the log history
   every detected fall stored in database."/></TextBlock>
9                          <TextBlock x:Name="textBlock1"
   HorizontalAlignment="Left" Margin="392,138,0,0"
   TextWrapping="Wrap" VerticalAlignment="Top" FontSize="13.333"
   Height="125" Width="12"><Run Text="a."/><LineBreak/><Run
   Text="b."/><LineBreak/><Run/><LineBreak/><Run
   Text="c."/><LineBreak/><Run/><LineBreak/><Run
   Text="d."/></TextBlock>
```

Kode Sumber 4.25 Menampilkan Tab Tentang (a)

```
10                      <TextBlock x:Name="textBlock2"
   HorizontalAlignment="Left" Margin="10,320,0,0"
   TextWrapping="Wrap" VerticalAlignment="Top" Height="78"
   Width="182"><Run FontWeight="Bold" FontSize="14.667"
   IsEnabled="False"/><LineBreak/><Run FontWeight="Bold"
   FontSize="14.667" IsEnabled="False" Text="About
   Developer"/><LineBreak FontSize="16"/><Run FontSize="13.333"
   Text="Fahmy Thoriqul Haq"/><LineBreak FontSize="16"/><Run
   FontSize="13.333" Text="fahmythor@gmail.com"/><LineBreak
   FontSize="16"/><Run FontSize="13.333" Text=" "/></TextBlock>
11                      <TextBlock x:Name="textBox"
   HorizontalAlignment="Left" Height="17" Margin="632,391,0,0"
   TextWrapping="Wrap" Text="Copyright © 2016 Mahasiswa Teknik
   Informatika" VerticalAlignment="Top" Width="259"/>
12                  </Grid>
13
14              </Controls:MetroTabItem>
```

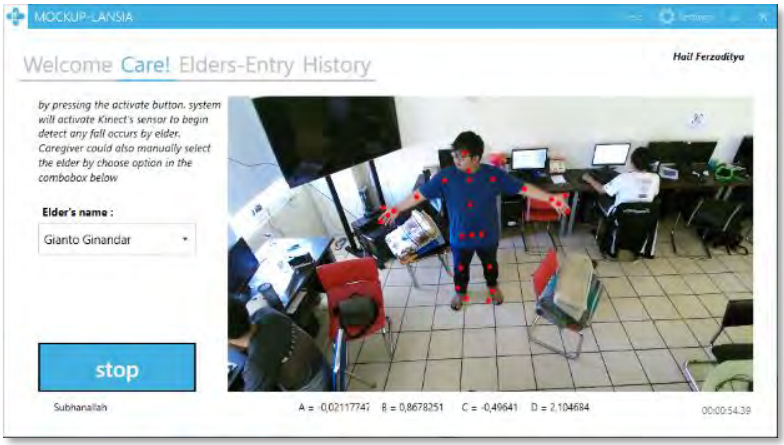
Kode Sumber 4.26 Menampilkan Tab Tentang (b)

4.4.4. Implementasi Jendela Utama – Tab Awasi

Tab "Awasi" mengalami perubahan menjadi *tab* "Care!" dengan tombol *activate* untuk mengaktifkan pendeteksian jatuh. Sebelum tombol tersebut ditekan maka Kinect belum menangkap gambar dan *view* untuk tangkapan Kinect masih berwarna abu-abu. Untuk lebih detail mengenai *tab* ini dapat dilihat pada Gambar 4.13

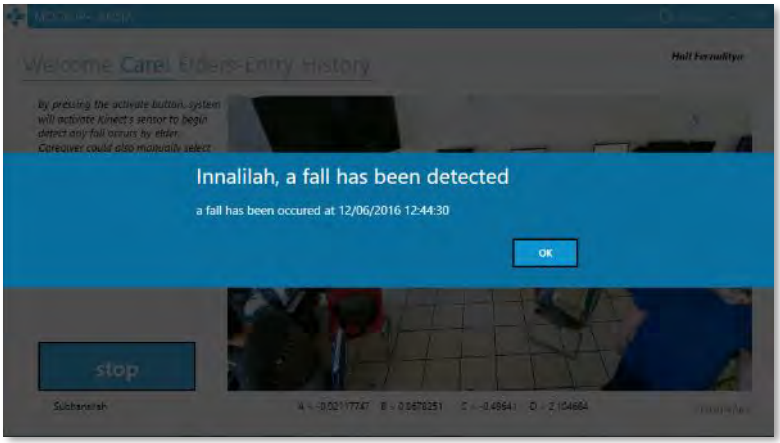
Gambar 4.13 Jendela Utama – Tab Awasi

Ketika tombol *activate* ditekan, Kinect akan aktif dan menampilkan *view* tangkapan Kinect. Selain itu, tombol *activate* akan menjadi tombol *stop* ketika Kinect mulai aktif dimana berfungsi untuk mematikan Kinect dan mode pendeteksian jatuh jika ditekan. Selain tombol utama tersebut, pada *tab* "Care!" terdapat *stopwatch* di bagian pojok kanan bawah antarmuka. *Stopwatch* akan aktif juga dengan ditekannya tombol *activate.* Sebaliknya, ketika tombol *stop* ditekan, *stopwatch* akan terinisialisasi lagi mulai dari 0. Untuk lebih jelas dapat dilihat pada Gambar 4.14.



Gambar 4.14 Jendela Utama – Tab Awasi saat Mode Aktif

Ketika Kinect mendeteksi tubuh maka pendeteksian jatuh seketika akan aktif untuk tubuh yang terdeteksi Kinect. Kemudian aplikasi akan mulai pemrosesan tiap *frame* hasil tangkapan Kinect dengan algoritma pendeteksi jatuh untuk mendeteksi apakah kejadian ajtuh terdeteksi atau sebaliknya. Ketika aplikasi mendeteksi jatuh, jendela utama akan memunculkan *popup* yang menampilkan pesan informasi telah terjadi jatuh disertai dengan waktu terjadinya. Untuk lebih detailnya dapat dilihat pada Gambar 4.15.



Gambar 4.15 Popup Ketika Terdeteksi Jatuh

Untuk kode sumber dari *tab* "Awasi" dapat dilihat pada Kode Sumber 4.27 dan Kode Sumber 4.28.

```
1  <Controls:MetroTabItem Header="Care!">
2                      <Grid Margin="10,5,-10,-13">
3
4                          <Grid VerticalAlignment="Top"
   HorizontalAlignment="Left" Height="360" Width="640"
   Margin="241,10,0,0" Background="#FFA6A6A6"/>
5                          <Image x:Name="image"
   HorizontalAlignment="Left" Height="360" Margin="241,10,0,0"
   VerticalAlignment="Top" Width="640"/>
6                          <Canvas Name="bodyCanvas"
   HorizontalAlignment="Left" Height="360" Margin="241,10,0,0"
   VerticalAlignment="Top" Width="640"/>
7
8                          <Button Style="{DynamicResource
   AccentedSquareButtonStyle}" x:Name="button" Content="ACTIVATE"
   HorizontalAlignment="Left" Margin="10,310,0,0"
   VerticalAlignment="Top" Width="192" Height="60"
   FontSize="26.667" Click="button_Click"/>
9                          <TextBox x:Name="TulisanStopWatch"
   HorizontalAlignment="Center" Height="23"
   Margin="757,380,10,10" TextWrapping="Wrap" Text="00:00:00.00"
   VerticalAlignment="Center" Width="124" TextAlignment="Right"
   BorderBrush="Transparent" IsEnabled="False"/>
```

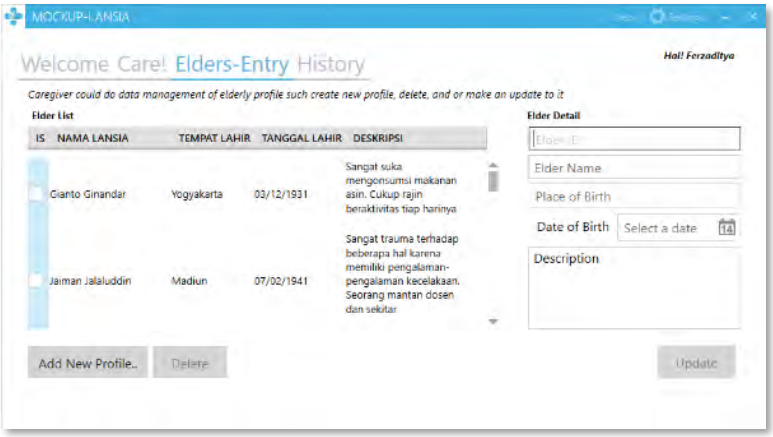
Kode Sumber 4.27 Menampilkan Tab Awasi (a)

```
10                      <Label x:Name="label_Ax" Content="Ax"
  HorizontalAlignment="Left" Margin="322,375,0,0"
  VerticalAlignment="Top" Width="96"/>
11                      <Label x:Name="label_By" Content="By"
  HorizontalAlignment="Left" Margin="423,375,0,0"
  VerticalAlignment="Top" Width="93"/>
12                      <Label x:Name="label_Cz" Content="Cz"
  HorizontalAlignment="Left" Margin="521,375,0,0"
  VerticalAlignment="Top" Width="79"/>
13                      <Label x:Name="label_D" Content="D"
  HorizontalAlignment="Left" Margin="605,375,0,0"
  VerticalAlignment="Top" Width="82"/>
14                      <Label x:Name="label_Fall"
  Content="Fall?" HorizontalAlignment="Left" Margin="24,375,0,0"
  VerticalAlignment="Top" Width="178"/>
15                      <ComboBox x:Name="comboBox_lansia"
  HorizontalAlignment="Left" Margin="10,164,0,0"
  VerticalAlignment="Top" Width="192" Height="39"
  Loaded="comboBox_lansia_Loaded" FontSize="14.667"
  MaxDropDownHeight="150"/>
16                      <Label x:Name="label" Content="Elder's
  name :" HorizontalAlignment="Left" Margin="10,136,0,0"
  VerticalAlignment="Top" FontSize="13.333" FontWeight="Bold"/>
17                      <TextBlock x:Name="textBlock4"
  HorizontalAlignment="Left" Margin="10,10,0,0"
  TextWrapping="Wrap" VerticalAlignment="Top" Height="126"
  Width="226" FontSize="13.333" FontStyle="Italic" Text="by
  pressing the activate button. system will activate Kinect's
  sensor to begin detect any fall occurs by elder. Caregiver
  could also manually select the elder by choose option in the
  combobox below"/>
18
19                  </Grid>
20              </Controls:MetroTabItem>
```

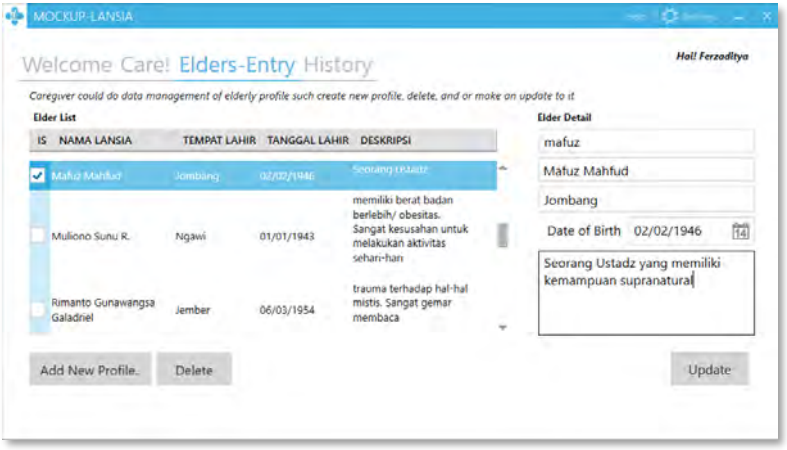
Kode Sumber 4.28 Menampilkan Tab Awasi (b)

4.4.5. Implementasi Jendela Utama – Tab Data Lansia

Tab "Data Lansia" juga mengalami perubahan menjadi *tab* "Elders-Entry". Pada *tab* ini memuat tabel yang berisi data-data lansia yang telah tersimpan pada *database*. Selain itu, ada juga kotak masukan pada *tab* bagian kanan yang memeberikan info detail dari data lansia pada tabel yang sedang terpilih/ aktif. untuk lebih detail mengenai *tab* ini dapat dilihat pada Gambar 4.16 dan Gambar 4.17.

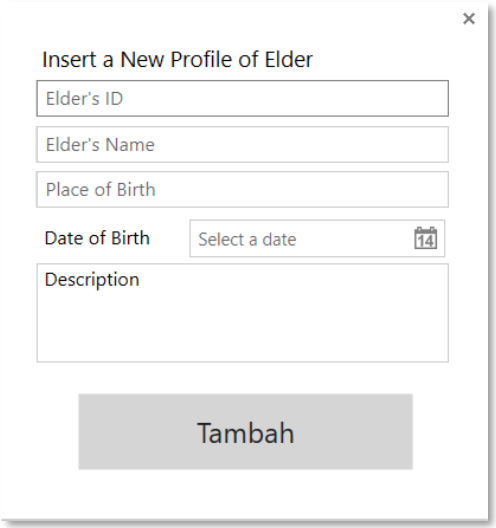


Gambar 4.16 Jendela Utama – Tab Data Lansia



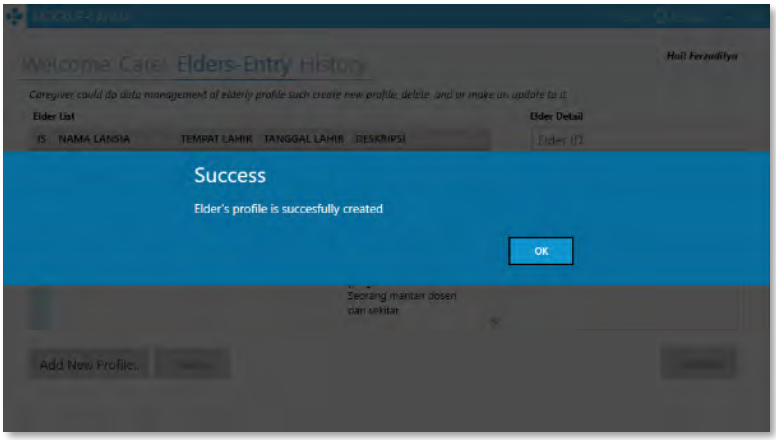
Gambar 4.17 Tab Data Lansia Dengan Data Lansia Terpilih

Selain itu, terdapat tiga tombol fungsionalitas juga pada *tab* ini. Tombol-tombol tersebut tidak lain adalah tombol *insert/add,* tombol *delete,* dan tombol *update*. Tombol *insert/add* akan menampikan sebuah jendela tambahan baru yaitu jendela tambah data lansia seperti ditunjukkan pada Gambar 4.18.



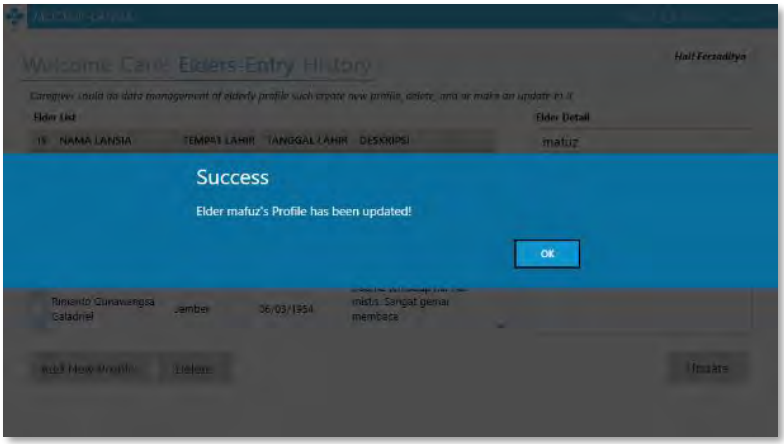
Gambar 4.18 Jendela Tambah Lansia

Setelah mengisi semua kotak masukan untuk data lansia yang hendak ditambahkan dan menekan tombol *insert* akan muncul *popup* berhasil seperti ditunjukkan pada Gambar 4.19

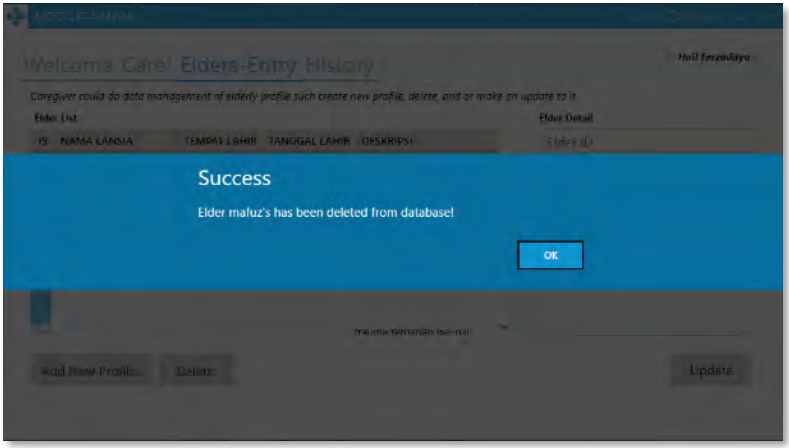


Gambar 4.19 Popup Berhasil Menambah Data Lansia

Begitu pula dengan tombol *update* dan *delete*. Yang akan melakukan perubahan atau penghapusan pada data lansia yang sedang terpilih pada tabel daftar lansia di bagian kiri. *Popup* akan muncul seperti pada Gambar 4.20 dan Gambar 4.21.



Gambar 4.20 Popup Berhasil Mengubah Data Lansia



Gambar 4.21 Popup Berhasil Menghapus Data Lansia

Untuk kode sumber yang telah diimplementasi untuk *tab* ini ditunjukkan pada Kode Sumber 4.29, Kode Sumber 4.30, dan Kode Sumber 4.31.

```
<Controls:MetroTabItem Header="Elders-Entry">
                    <Grid>
                        <Grid.Resources>
                            <ResourceDictionary>

<ResourceDictionary.MergedDictionaries>
                                    <ResourceDictionary
Source="pack://application:,,,/MahApps.Metro;component/Styles/
FlatButton.xaml" />

</ResourceDictionary.MergedDictionaries>
                            </ResourceDictionary>
                        </Grid.Resources>
                        <DataGrid x:Name="datagrid_Lansia"
Margin="10,54,307,104" ItemsSource="{Binding}"
SelectionChanged="datagrid_Lansia_selectionChanged"
Style="{StaticResource AzureDataGrid}"
CanUserResizeColumns="False" CanUserResizeRows="False"
AutoGeneratedColumns="datagrid_Lansia_AutoGeneratedColumns"
FontSize="13">
                            <DataGrid.Columns>
                                <DataGridCheckBoxColumn
ElementStyle="{DynamicResource MetroDataGridCheckBox}"

EditingElementStyle="{DynamicResource MetroDataGridCheckBox}"
                                    Header="is"
                                    Binding="{Binding
RelativeSource={RelativeSource AncestorType=DataGridRow},
Path=IsSelected, Mode=OneWay}"
                                    />
                            </DataGrid.Columns>
                        </DataGrid>
                        <TextBox x:Name="txtbox_IDLansia"
Controls:TextBoxHelper.Watermark="Elder ID"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="0,54,9,324" FontSize="16" TabIndex="1"
HorizontalAlignment="Right" Width="261" />
                        <TextBox x:Name="txtbox_NamaLansia"
Controls:TextBoxHelper.Watermark="Elder Name"
Controls:TextBoxHelper.SelectAllOnFocus="True"
Margin="0,89,9,289" FontSize="16" TabIndex="2"
HorizontalAlignment="Right" Width="262" />
```

Kode Sumber 4.29 Menampilkan Tab Data Lansia (a)

```
22                          <TextBox x:Name="txtbox_TmpLahir"
   Controls:TextBoxHelper.Watermark="Place of Birth"
   Controls:TextBoxHelper.SelectAllOnFocus="True"
   Margin="0,124,9,252" FontSize="16" TabIndex="2"
   HorizontalAlignment="Right" Width="261" />
23                          <RichTextBox
   x:Name="richTextBox_deskripsi" HorizontalAlignment="Right"
   Height="102" Margin="0,202,9,0" VerticalAlignment="Top"
   Width="262" FontSize="16">
24                              <FlowDocument>
25                                  <Paragraph>
26                                      <Run Text="Description"/>
27                                  </Paragraph>
28                              </FlowDocument>
29                          </RichTextBox>
30                          <Button x:Name="btn_insert"
   Content="Add New Profile.." HorizontalAlignment="Left"
   Margin="10,325,0,0" VerticalAlignment="Top" Width="148"
   FontSize="16" Height="39" Click="btn_insert_Click"
   TabIndex="3"/>
31                          <Button x:Name="btn_update"
   Content="Update" HorizontalAlignment="Left"
   Margin="786,324,0,0" VerticalAlignment="Top" Width="95"
   FontSize="16" Height="40" TabIndex="3"
   Click="btn_update_Click"/>
32                          <Button x:Name="btn_delete"
   Content="Delete" HorizontalAlignment="Left"
   Margin="164,325,0,0" VerticalAlignment="Top" Width="91"
   FontSize="16" Height="39" TabIndex="3"
   Click="btn_delete_Click"/>
33                          <DatePicker
   x:Name="datepick_TglLahirLansia" HorizontalAlignment="Right"
   Margin="0,161,9,0" VerticalAlignment="Top" Height="34"
   Width="152" FontSize="16"/>
34                          <Label x:Name="label1" Content="Date
   of Birth" HorizontalAlignment="Right" Margin="0,161,166,213"
   VerticalAlignment="Center" FontSize="16" Height="34"/>
35                          <TextBlock x:Name="textBlock3"
   HorizontalAlignment="Left" Margin="10,4,0,0"
   TextWrapping="Wrap" Text="Caregiver could do data management
   of elderly profile such create new profile, delete, and or
   make an update to it" VerticalAlignment="Top" Width="859"
   FontSize="13.333" FontStyle="Italic"/>
36                          <Label x:Name="label2" Content="Elder
   Detail" HorizontalAlignment="Left" Margin="620,27,0,0"
   VerticalAlignment="Top" FontWeight="Bold"/>
```

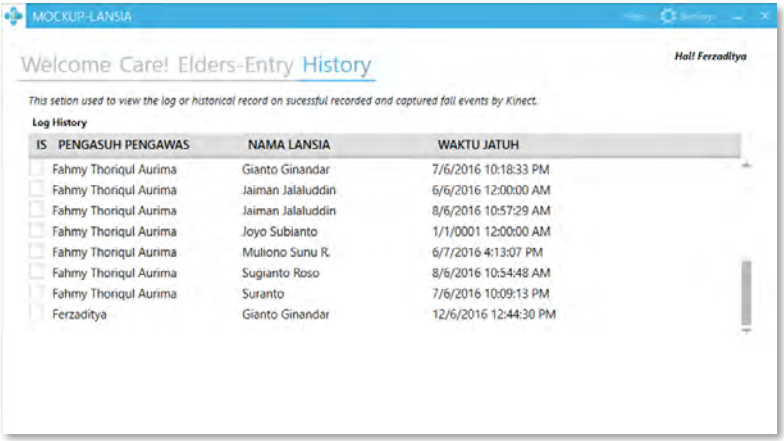
Kode Sumber 4.30 Menampilkan Tab Data Lansia (b)

```
37                          <Label x:Name="label2_Copy"
   Content="Elder List" HorizontalAlignment="Left"
   Margin="10,27,0,0" VerticalAlignment="Top" FontWeight="Bold"/>
38                      </Grid>
39                  </Controls:MetroTabItem>
```

Kode Sumber 4.31 Menampilkan Tab Data Lansia (c)

4.4.6. Implementasi Jendela Utama – Tab Historis

Pada *tab* "Historis" terjadi juga perubahan nama menjadi *tab* "History" dengan konten utama berisi tabel yang menampilkan *log* kejadian jatuh yang terdeteksi dan tersimpan pada *database* setelah pendeteksian jatuh terjadi. Untuk lebih detail tampilan pada jendela ini, dapat dilihat pada Gambar 4.22.



Gambar 4.22 Jendela Utama – Tab Histori

Sedangkan kode sumber untuk tampilan *tab* ini dapat dilihat pada Kode Sumber 4.32 dan Kode Sumber 4.33.

```
1  <Controls:MetroTabItem Header="History">
2                      <Grid>
3                          <Grid.ColumnDefinitions>
4                              <ColumnDefinition/>
5                          </Grid.ColumnDefinitions>
```

Kode Sumber 4.32 Menampilkan Tab Histori (a)

```
6                       <DataGrid x:Name="datagrid_Jatuh"
Margin="10,59,10,99" ItemsSource="{Binding}"
CanUserResizeColumns="False" CanUserResizeRows="False"
FontSize="14.667"
AutoGeneratedColumns="datagrid_Jatuh_AutoGeneratedColumns">
7                           <DataGrid.Columns>
8
9                               <DataGridCheckBoxColumn
ElementStyle="{DynamicResource MetroDataGridCheckBox}"
10
EditingElementStyle="{DynamicResource MetroDataGridCheckBox}"
11                                      Header="is"
12                                      Binding="{Binding
RelativeSource={RelativeSource AncestorType=DataGridRow},
Path=IsSelected, Mode=OneWay}"
13                                      />
14                          </DataGrid.Columns>
15
16                      </DataGrid>
17                      <TextBlock x:Name="textBlock4_Copy"
HorizontalAlignment="Left" Margin="10,10,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Height="18"
Width="871" FontSize="13.333" FontStyle="Italic" Text="This
setion used to view the log or historical record on sucessful
recorded and captured fall events by Kinect."/>
18                      <Label x:Name="label3" Content="Log
History" HorizontalAlignment="Left" Margin="10,33,0,0"
VerticalAlignment="Top" FontWeight="Bold"/>
19
20                  </Grid>
21
22              </Controls:MetroTabItem>
```

Kode Sumber 4.33 Menampilkan Tab Histori (b)

BAB V
PENGUJIAN DAN EVALUASI

Bab ini akan membahas dan menjelaskan pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian adalah pengujian fungsionalitas proses pada aplikasi yang sesuai dengan penjabaran sebelumnya pada Bab III yang intinya dapat mendeteksi gerakan jatuh pada seseorang.

5.1. Lingkungan Pengujian

Lingkungan pengujian aplikasi pada pengerjaan tugas akhir kali ini meliputi perangkat keras dan perangkat lunak sebagai berikut:

Prosesor	: Prosesor Intel® Core™ i7-3610QM CPU @ 2.30 GHz (8 CPUs)
RAM	: 4096 MB
Jenis *Device*	: *Notebook*
Sistem Operasi	: Windows 10 Pro 64-bit

5.2. Skenario dan Hasil Pengujian

Pada subbab ini akan dijelaskan mengenai skenario pengujian yang dilakukan pada sistem. Pengujian yang dilakukan terdiri dari pengujian fungsionalitas dan pengujian akurasi. Pengujian fungsionalitas menggunakan metode kotak hitam (*black box)* yang menekankan pada kesesuaian antara hasil keluaran sistem dengan hasil yang diharapkan. Pengujian akurasi dilakukan dengan menggunakan peraga yang akan melakukan adegan sesuai skenario jatuh dan tidak jatuh kemudian melihat bagaimana tanggapan kebenaran pendeteksi jatuh.

5.2.1. Pengujian Fungsionalitas

Pengujian fungsionalitas aplikasi dilakukan secara mandiri dengan melakukan skenario yang sama dengan perancangan proses aplikasi sebagai tolok ukur keberhasilan pengujian, dan mengacu pada deskripsi proses yang sebelumnya telah dijelaskan pada Bab

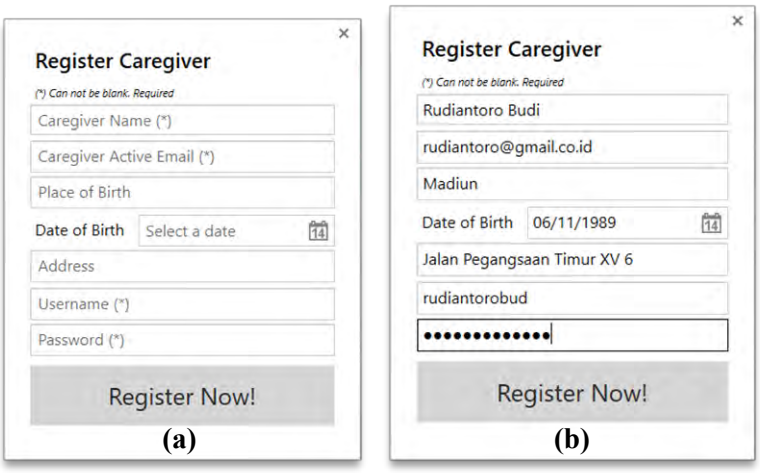
III. Pengujian pada kebutuhan fungsionalitas dapat dijabarkan pada subbab berikut.

5.2.1.1. Pengujian Mendaftar Pada Sistem

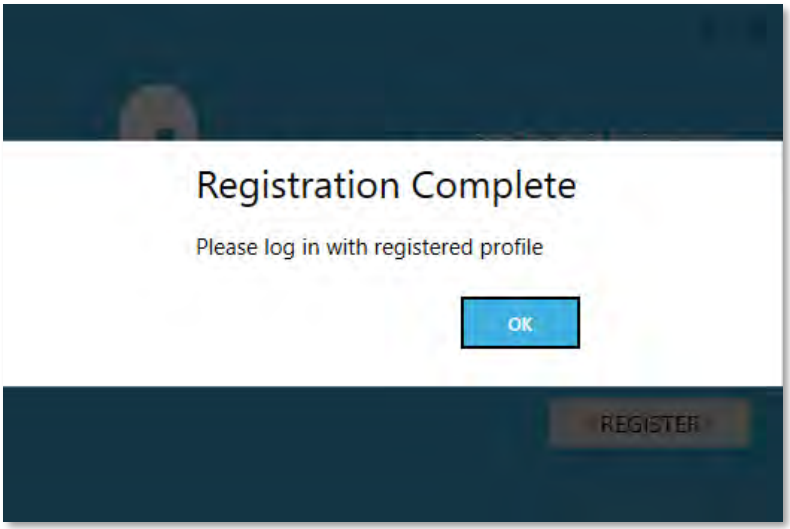
Pengujian mendaftar pada sistem dilakukan untuk menguji apakah informasi data profil pengasuh yang dimasukkan dapat tersimpan ke dalam *database*. Pendaftaran dilakukan pengasuh ketika belum memiliki akun terdaftar pada aplikasi.

Tabel 5.1 Pengujian Fungsionalitas Proses Mendaftar Pada Sistem

Nomor	SP-Proses-1.1
Referensi Proses	Proses 1.1 Mendaftar Pada Sistem
Nama	Pengujian Proses Mendaftar Pada Sistem
Tujuan	Menguji apakah aplikasi dapat menyimpan data profil pengasuh pada *database*
Kondisi Awal	Sistem menampilkan jendela registrasi.
Skenario 1 – Registrasi dengan data baru	
Skenario	Pengasuh memasukkan informasi data profil baru yang diminta untuk registrasi pada kontak masukan pada jendela registrasi, kemudian melakukan registrasi.
Kondisi Akhir yang Diharapkan	Jendela utama menampilkan *popup* bahwa registrasi telah berhasil dan Data masukan registrasi berhasil tersimpan pada *database*.
Hasil Pengujian	100% Berhasil. Sistem menyimpan data masukan profil pengasuh dan menampilkan *popup* bahwa pengasuh telah berhasil terdaftar pada sistem.
Skenario 2 – Registrasi dengan *username* yang telah terdaftar	
Skenario	Pengasuh memasukkan informasi data profil dengan *username* pengasuh yang sebelumnya telah terdaftar pada sistem, kemudian melakukan registrasi.
Kondisi Akhir yang Diharapkan	Jendela utama menampilkan *popup* bahwa registrasi gagal dan memberikan pesan terjadi kesalahan bahwa *username* telah digunakan.
Hasil Pengujian	100% Berhasil. Sistem menampilkan *popup* bahwa proses registrasi gagal karena *username* telah digunakan dan terdaftar pada sistem.

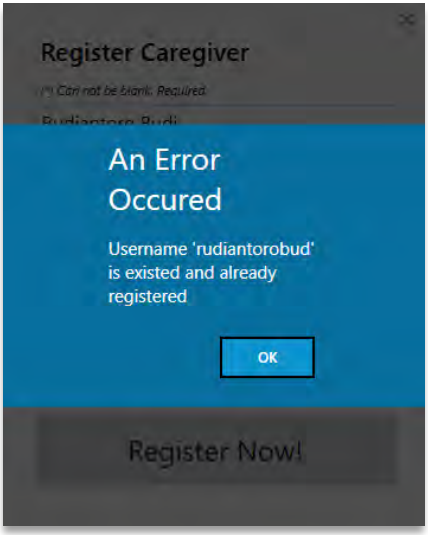


Gambar 5.1 Jendela Registrasi Sebelum Terisi Data (a) dan Ketika Telah Terisi Data Registrasi (b)



Gambar 5.2 Popup Pesan Berhasil Registrasi

Detail dari pengujian menggunakan skenario 1 dapat dilihat pada Tabel 5.1. bagian skenario 1. Sedangkan keluaran yang dihasilkan oleh sistem dapat dilihat pada Gambar 5.1 dimana aplikasi menampilkan jendela registrasi dan meminta masukan dari pengasuh untuk memasukkan data registrasi, kemudian pada Gambar 5.2 aplikasi berhasil menampilkan *popup* pesan berhasil registrasi. Sedangkan untuk detail pada pengujian skenario 2 dapat dilihat pada Tabel 5.1 bagian skenario 2 dengan detail keluaran aplikasi dalam menampilkan *popup* kesalahan yang dapat dilihat pada Gambar 5.3 berikut.



Gambar 5.3 Popup Pesan Kesalahan Registrasi

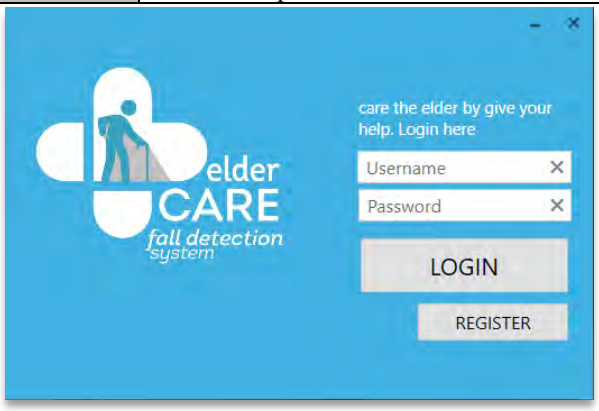
5.2.1.2. Pengujian Masuk Dalam Sistem

Pengujian masuk dalam sistem dilakukan untuk mengecek dan menguji apakah aplikasi dapat mencari kesesuaian masukan *username* dan *password* pada kotak masukan dengan data registrasi yang telah tersimpan dalam *database.* Hal ini merupakan proses otentikasi sederhana untuk menentukan apakah kombinasi *username* dan *password* tersebut dapat memberikan hak akses pada

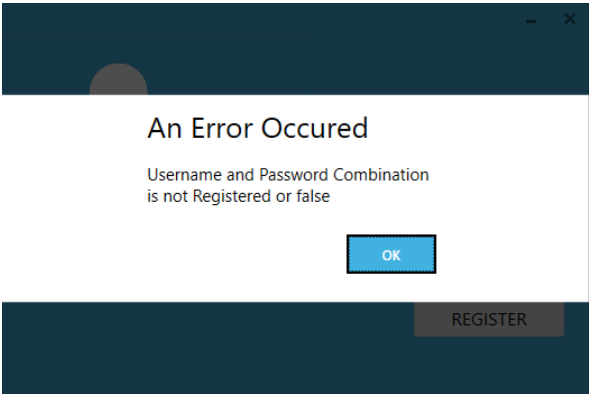
aplikasi atau tidak. Masuk dalam sistem dilakukan pada jendela *login* yang merupakan jendela pertama kali ketika sistem berjalan.

Tabel 5.2 Pengujian Fungsionalitas Proses Masuk Dalam Sistem

Nomor	SP-Proses-1.2
Referensi Proses	Proses 1.2 Masuk Dalam Sistem
Nama	Pengujian Proses Masuk Dalam Sistem
Tujuan	Menguji apakah aplikasi dapat melakukan otetikasi sederhana dengan mengecek *username* dan *password* masukan dengan data registrasi pada *database.*
Kondisi Awal	Sistem menampilkan jendela *login*.
Skenario	Pengasuh memasukkan *username* dan *password* pada kotak masukan kemudian melakukan *login.*
Kondisi Akhir yang Diharapkan	1. Jika Otentikasi benar, aplikasi akan membuka jendela utama aplikasi. 2. Jika Otentikasi salah, aplikasi akan memberikan *popup* pesan yang memberitahu adanya kesalahan dalam kombinasi *user* dan *password.*
Hasil Pengujian	100% Berhasil. Berhasil menjalankan otentikasi sederhana dengan pengecekan kombinasi *username* dan *password* pada data registrasi yang telah tersimpan di *database.*

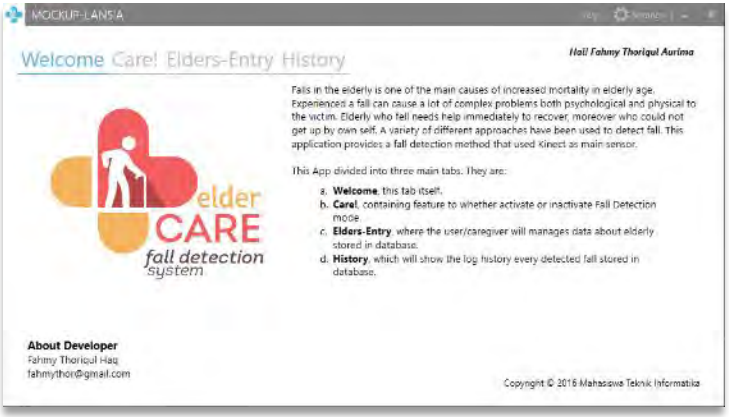


Gambar 5.4 Kondisi Awal Jendela Login



Gambar 5.5 Popup Pesan Kesalahan Login

Tampilan pengujian jendela masuk dapat dilihat pada Gambar 5.4. Sedangkan detail hasil pengujian dapat dilihat pada Tabel 5.2 dan Gambar 5.5 dimana ketika otentikasi gagal, maka akan muncul *popup* kesalahan yang berisi pesan bahwa kombinasi *username* dan *password* yang dimasukkan tidak tepat atau belum terdaftar pada sistem. Sedangkan ketika otentikasi berhasil, aplikasi akan langsung menuju ke jendela utama seperti ditunjukkan pada Gambar 5.6.



Gambar 5.6 Jendela Utama

5.2.1.3. Pengujian Mengaktifkan Kinect

Pengujian mengaktifkan Kinect dilakukan untuk menguji apakah aplikasi mampu untuk memanggil Kinect dan mengaktifkan Kinect melalui tombol pemicu pada aplikasi.

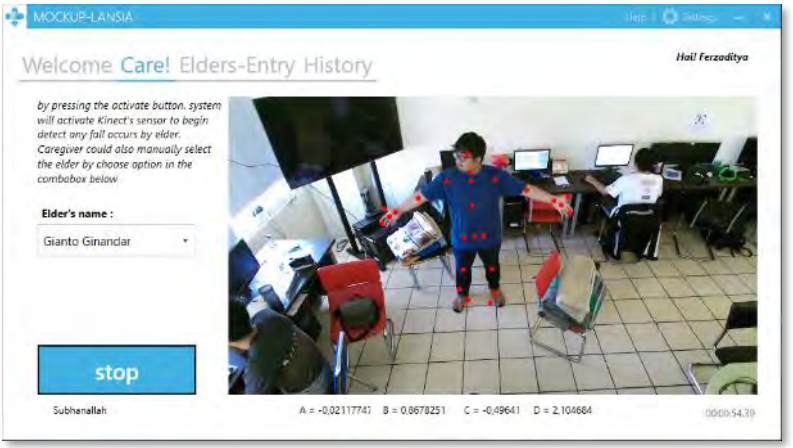
Tabel 5.3 Pengujian Fungsionalitas Proses Mengaktifkan Kinect

Nomor	SP-Proses-1.3
Referensi Proses	Proses 1.3 Mengaktifkan Kinect
Nama	Pengujian Proses Mengaktifkan Kinet
Tujuan	Menguji apakah aplikasi dapat mengaktifkan Kinect melalui pemicu pada aplikasi
Kondisi Awal	Kinect belum aktif
Skenario	Pengasuh menekan tombol pemicu untuk mengaktifkan Kinect.
Kondisi Akhir yang Diharapkan	Kinect akan berhasil menyala dan dalam posisi aktif. Selain itu, aplikasi akan menampilkan hasil tangkapan Kinect secara *realtime*.
Hasil Pengujian	100% Berhasil. Aplikasi berhasil mengaktifkan Kinect melalui tombol pemicu.

Gambar 5.7 Kinect Dalam Keadaan Aktif

Seperti dapat dilihat pada Gambar 5.7, Kinect akan berada dalam keadaan aktif ketika tombol pemicu pada jendela utama aplikasi ditekan. Sedangkan pada jendela utama sendiri, aplikasi

akan menampilkan hasil tangkapan Kinect berupa *frame* gambar berwarna seperti ditunjukkan pada Gambar 5.8 berikut.



Gambar 5.8 Tangkapan Kinect Ketika Aktif

5.2.1.4. Pengujian Proses Mendeteksi Jatuh

Pengujian mendeteksi jatuh adalah inti dari aplikasi ini. Dimana aplikasi harus mampu untuk mendeteksi kejadian jatuh yang dialami oleh tubuh yang tertangkap oleh sensor Kinect. Pengujian ini dilakukan untuk mengetahui apakah aplikasi mampu untuk melakukan hal tersebut dengan baik. Selain mendeteksi jatuh, aplikasi juga akan diuji apakah dapat menyimpan rekam historis informasi jatuh di *database*.

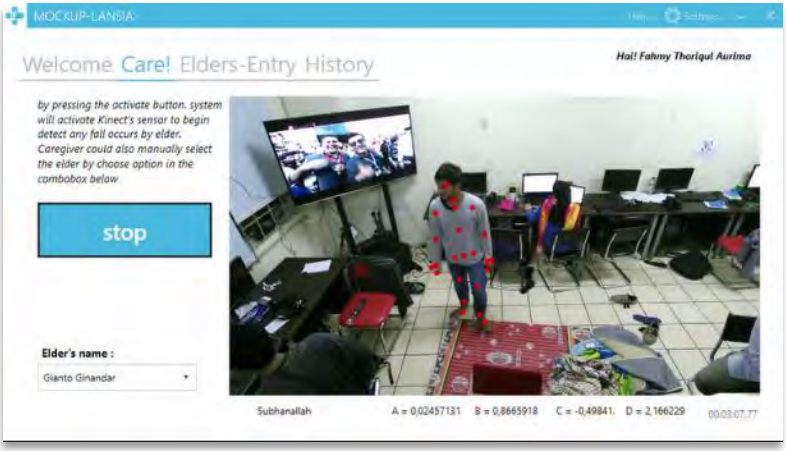
Tabel 5.4 Pengujian Fungsionalitas Proses Mendeteksi Jatuh (a)

Nomor	SP-Proses-1.4
Referensi Proses	Proses 1.4 Mendeteksi Jatuh
Nama	Pengujian Proses Mendeteksi Jatuh
Tujuan	Menguji apakah aplikasi dapat mendeteksi jatuh dengan baik atau tidak sekaligus menyimpan rekam dalam *database*.
Kondisi Awal	Kinect diaktifkan

Tabel 5.5 Pengujian Fungsionalitas Proses Mendeteksi Jatuh (b)

Skenario	Mode pengawasan/ pendeteksian jatuh dalam keadaan aktif dan terjadi jatuh pada tubuh yang tertangkap Kinect.
Kondisi Akhir yang Diharapkan	Muncul *popup* yang memberikan infomasi telah terjadi jatuh dan data rekam tersebut tersimpan dalam *database*.
Hasil Pengujian	100% Berhasil. Aplikasi berhasil mendeteksi jatuh dan memberikan *popup* sekaligus menyimpan data rekam jatuh di *database*.

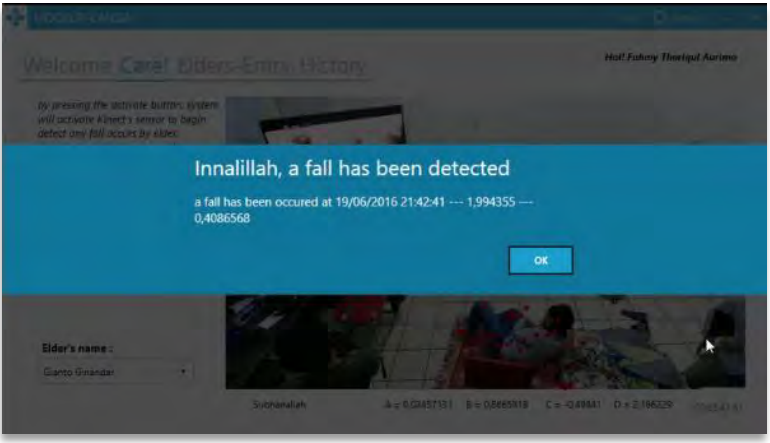
Seperti dapat dilihat pada Tabel 5.4 dan Tabel 5.5, kondisi awal adalah ketika Kinect telah aktif, maka Kinect akan memulai untuk mendeteksi tubuh peraga seperti juga dapat dilihat pada Gambar 5.9 berikut.



Gambar 5.9 Kinect Aktif Mulai Mendeteksi Tubuh

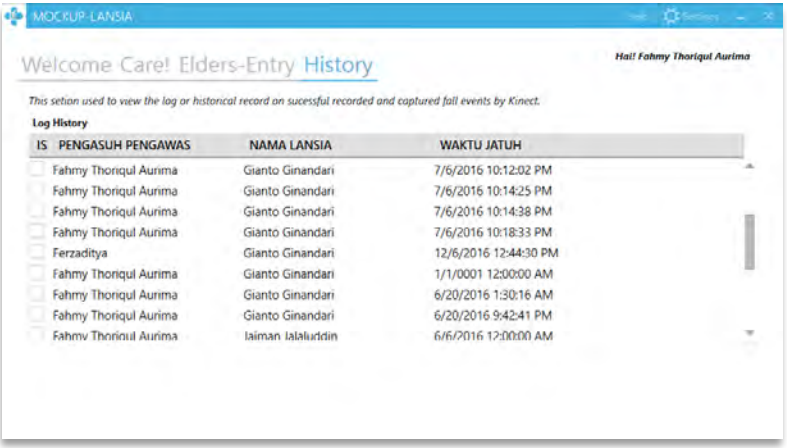
Setelah mendeteksi tubuh, maka aplikasi akan langsung melakukan pemrosesan tiap *frame* tubuh tangkapan Kinect pada algoritma *decision tree* yang telah diimplementasikan sesuai subbab 4.3 sebelumnya. Ketika tidak terdeteksi jatuh, maka tidak aka nada sesuatu yang terjadi dan Kinect tetap akan mengikuti pendeteksian terhadap tubuh yang sedang terdeteksi. Setelah itu, ketika jatuh

terjadi, maka aplikasi akan mengeluarkan keluaran *popup* seperti pada Gambar 5.10 berikut.



Gambar 5.10 Popup Pesan Deteksi Jatuh

Setelah itu, pengujian berlanjut untuk melihat apakah data jatuh berhasil disimpan pada *database*, selengkapnya dapat dilihat pada Gambar 5.11 berikut



Gambar 5.11 Data Jatuh Tersimpan Pada Database

5.2.1.5. Pengujian Mengelola Data Lansia

Pengujian mengelola data lansia ditujukan untuk menguji kemampuan aplikasi untuk melakukan pengelolaan pada data yang benar sesuai dengan yang ditentukan oleh pengasuh. Selain itu, mengecek apakah aplikasi mampu melakukan pengelolaan data sederhana seperti menambah, menggubah, dan menghapus data lansia dalam *database*.

Tabel 5.6 Pengujian Fungsionalitas Mengelola Data Lansia (a)

Nomor	SP-Proses-1.5
Referensi Proses	Proses 1.5 Mengelola Data Lansia
Nama	Pengujian Proses Mengelola Data Lansia
Tujuan	Menguji fitur untuk pengasuh dalam mengelola data lansia pada aplikasi.
Skenario I : Menguji fitur menambah data lansia	
Kondisi Awal	Sistem menampilkan jendela tambah data lansia baru dan data lansia belum ada
Skenario	Pengasuh memasukkan semua data profil lansia baru yang hendak ditambahkan pada aplikasi.
Kondisi Akhir yang Diharapkan	Muncul *popup* yang memberikan informasi keberhasilan menambah data lansia baru. Data lansia baru berhasil tersimpan dalam *database* aplikasi.
Hasil Pengujian	100% Berhasil. Aplikasi berhasil menampilkan *popup* dan data berhasil tersimpan dalam *database*.
Skenario II : Menguji fitur mengubah/ meng-*update* data lansia	
Kondisi Awal	Data lansia masih belum berubah/ terbarukan dalam *database*.
Skenario	Pengasuh memilih data lansia yang hendak diperbarui dan memperbarui data-data lansia tersebut dengan mengisi kotak masukan yang ditampilkan aplikasi.
Kondisi Akhir yang Diharapkan	Muncul *popup* berhasil *update* data lansia dan data dalam *database* telah terbarukan.

Tabel 5.7 Pengujian Fungsionalitas Mengelola Data Lansia (b)

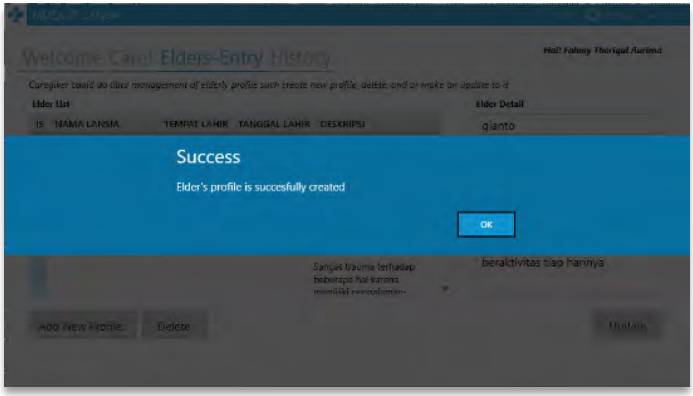
Hasil Pengujian	100% Berhasil. Aplikasi menampilkan *popup* berhasil *update* data dan perubahan tersimpan dalam *database*.
Skenario III : Menguji fitur menghapus data lansia	
Kondisi Awal	Data lansia masih ada dalam *database* aplikasi.
Skenario	Pengasuh memilih data lansia yang akan dihapus dari *database*.
Kondisi Akhir yang Diharapkan	Muncul *popup* berhasil hapus data lansia dari *database* dan data lansia tersebut terhapus dari *database*.
Hasil Pengujian	100% Berhasil. Aplikasi menampilkan *popup* data telah dihapus dan data tersebut terhapus dari *database*.

Detail hasil pengujian dapat dilihat pada Tabel 5.6 dan Tabel 5.7, dimana pengujian ini terdiri dari tiga skenario yakni skenario menambah, mengubah, dan menghapus data lansia.

Proses detail pengujian skenario pertama dapat dilihat lebih lanjut dipada Gambar 5.12 dan Gambar 5.14, dimana aplikasi menampilkan jendela untuk menambahkan data lansia baru.

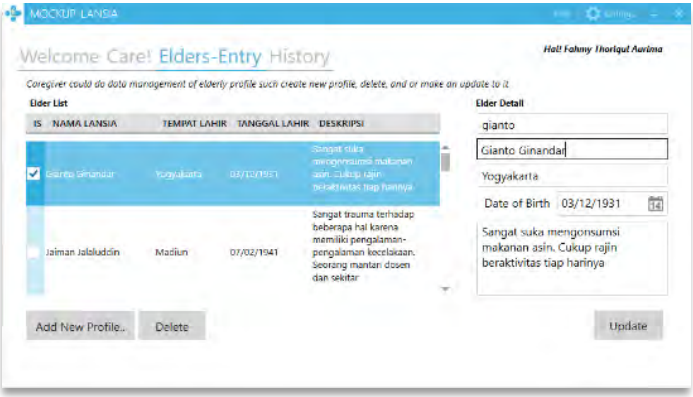
Gambar 5.12 Menambah Data Lansia Baru

Setelah data yang diminta telah terisi dan pengasuh melakukan penambahan maka dilanjutkan pada Gambar 5.13 yang menunjukkan aplikasi menampilkan *popup* pesan berhasil menambahkan data lansia baru pada *database*.

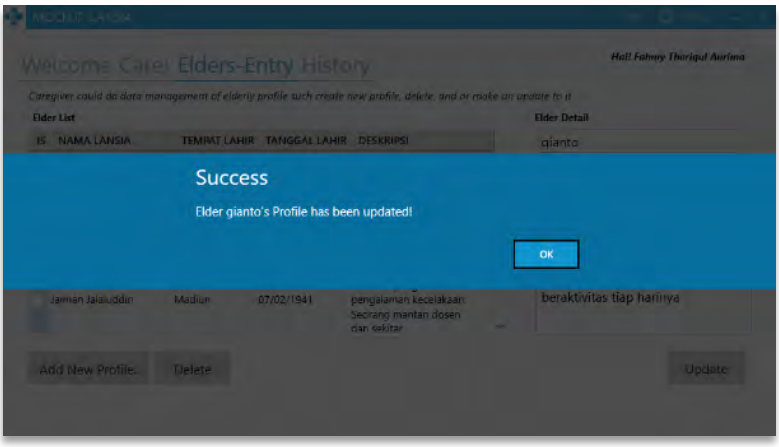


Gambar 5.13 Popup Pesan Berhasil Menambah Data Lansia Baru

Selanjutnya, pada skenario kedua detail hasil pengujian dapat dilihat pada Gambar 5.14 dan Gambar 5.15. Dimana pengasuh melakukan pengubahan pada data lansia terpilih dan kemudian dilanjutkan keluaran *popup* pesan berhasil melakukan pengubahan data lansia dan tersimpan pada *database*.

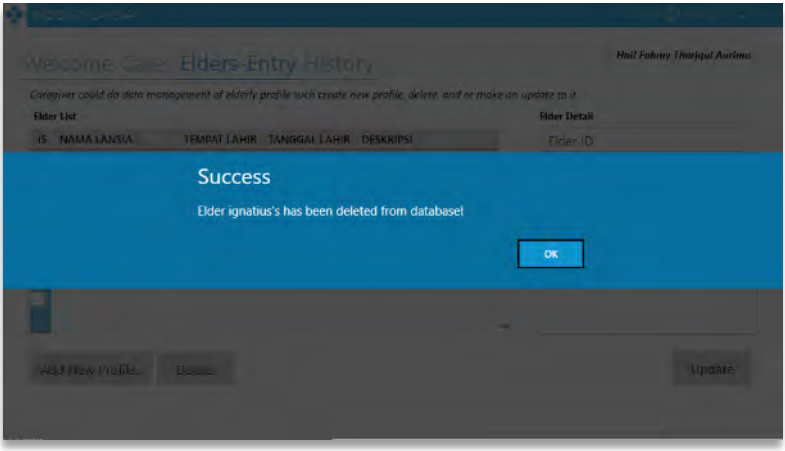


Gambar 5.14 Data Lansia Terpilih Tertampil



Gambar 5.15 Popup Pesan Berhasil Mengubah Data Lansia Terpilih

Pada skenario terakhir, yaitu skenario penghapusan data lansia terpilih dapat dilihat hasil keluaran pengujian yang dilakukan pada Gambar 5.16.



Gambar 5.16 Popup Pesan Berhasil Menghapus Data Lansia Terpilih

5.2.2. Pengujian Akurasi Deteksi Jatuh

Pengujian akurasi deteksi jatuh pada aplikasi dilakukan dengan mengacu pada implementasi algoritma sebagai acuan untuk diuji. Pengujian ini dilakukan untuk mengetahui apakah algoritma yang diimplementasikan berhasil mengklasifikasikan dan mendeteksi gerakan jatuh dengan tepat. Skenario pada pengujian ini meliputi beberapa skenario jatuh dan tidak jatuh yang akan diperagakan oleh beberapa orang peraga.

Uji coba ini menggunakan metode *confusion matrix* untuk menghitung akurasi. *Confusion matrix* atau juga dapat disebut sebagai *error matrix* adalah tabel khusus yang dapat memvisualisasikan performa sebuah algoritma. Metode ini memiliki beberapa nilai perthitungan. Diantaranya adalah:

- ***True Positive***
  Adalah ketika peragaan adalah benar jatuh dan terdeteksi sebagai jatuh oleh aplikasi.
- ***True Negative***
  Adalah ketika peragaan adalah tidak jatuh dan terdeteksi sebagai tidak jatuh oleh aplikasi
- ***False Positive***
  Adalah ketika peragaan adalah benar jatuh dan terdeteksi sebagai tidak jatuh oleh aplikasi
- ***False Negative***
  Adalah ketika peragaan adalah tidak jatuh dan terdeteksi sebagai jatuh oleh aplikasi

Yang kemudian dengan nilai-nilai tersebut tersebut nantinya akan didapatkan tingkat nilai akurasi dan banyak nilai-nilai penting, beberapa diantaranya adalah:

- Sensivitas adalah nilai proporsi kejadian yang benar-benar jatuh dalam populasi kejadian yang juga diidentifikasi sebagai kejadian jatuh oleh aplikasi. Dapat juga disebut sebagai *true positive rate*.
- Sebaliknya spesifitas adalah ukuran probabilitas identifikasi secara benar kejadian tidak jatuh yang

teridentifikasi sebagi tidak jatuh oleh sistem, atau bisa disebut *true negative rate*.

- Nilai *positive predictive value* (PPV), proporsi kejadian yang benar benar jatuh (*true positive*) di antara keseluruhan kejadian yang menunjukkan hasil deteksi jatuh.
- Nilai *negative predictive value* (NPV), persentase dari semua kejadian yang benar-benar tidak jatuh(*true negative*) diantara semua kejadian yang menunjukkan hasil deteksi negatif(tidak jatuh).

Sedangkan untuk skenario peragaan yang akan dilakukan pada uji coba kali ini adalah seagai berikut:

- Gerakan jatuh adalah ketika peraga/ aktor melakukan serangkaian gerakan dari aktivitas biasa/ statis dalam segala jenis kondisi berdiri menuju keadaan transisi dinamis dengan kecepatan yang tinggi kemudian berakhir ke kondisi statis yang posisi titik tertingginya lebih rendah dari titik tengah tubuh ketika berdiri seperti terbaring, jongkok, maupun berlutut.
- Gerakan tidak jatuh, Adalah ketika peraga melakukan gerakan aktivitas dalam keadaan biasa atau aktivitas non-jatuh. Aktivitas ini adalah gerakan-gerakan selain gerakan jatuh seperti telah dijelaskan pada poin sebelumnya. Gerakan-gerakan yang akan diujikan nantinya adalah gerakan-gerakan non-jatuh biasa dan gerakan yang menyerupai jatuh untuk lebih mengetahui seberapa besar *error rate* algoritma *decision tree* yang telah diimplementasikan nantinya. Beberapa diantaranya adalah meliputi:
  - Aktivitas dari berdiri/ berjalan menjadi duduk,
  - Aktivitas dari berdiri/ berjalan/ duduk menjadi posisi tidur(sejajar dengan lantai),
  - Aktivitas dari,
  - Aktivitas melompat,
  - Aktivitas berjalan,

- Aktivitas berlari,
- Aktivitas bangun dari tidur
- Aktivitas kombinasi yaitu kombinasi dari skenario yang telah disebut sebelumnya.

5.2.2.1. Pengujian Skenario Jatuh

Pada subbab ini akan dibahas lebih lanjut pengujian aplikasi dengan skenario jatuh yang telah disebutkan pada subbab sebelumnya. Uji coba dilakukan dengan peraga/ aktor yang akan menjalankan aktivitas skenario jatuh. Beberapa skenario jatuh adalah sebagai berikut:

- Jatuh ke depan, dimana gerakan jatuh dimulai dengan posisi tubuh condong dengan arah ke depan sepenuhnya hingga bagian atas tubuh akan berada sedekat mungkin dengan dataran.
- Jatuh ke samping, adalah ketika gerakan jatuh mengarah ke samping baik sisi kanan maupun kiri.
- Jatuh ke belakang, adalah ketika tubuh terjatuh dengan posisi tubuh condong ke belakang.
- Jatuh hingga berlutut, merupakan jatuh dengan arah sedikit ke depan hingga lutut bersentuhan dengan lantai.

Detail Hasil uji coba dapat dilihat pada Tabel 5.8 berikut.

Tabel 5.8 Pengujian Akurasi Aplikasi Dengan Skenario Jatuh

Skenario Jatuh	Jumlah Percobaan	True Positive (TP)	False Negative (FN)	% TP	% FN
Jatuh Ke Depan	15	15	0	100	0
Jatuh Ke Samping	15	15	0	100	0
Jatuh Ke Belakang	15	14	1	93,3	0,66
Jatuh Berlutut	15	14	1	93,3	0,66
Total	**60**	**58**	**2**	**386,6**	**1,76**

Dari tabel di atas, dapat diketahui beberapa poin diantaranya adalah skenario jatuh ke depan dan jatuh ke samping memiliki nilai *true positive* sebesar 100% dan nilai *false negative* sama besar 0%. Sedangkan untuk skenario jatuh ke belakang dan jatuh berlutut mengalami sekali kesalahan deteksi dari total 15 percobaan. Sehingga nilai *true positive* adalah sebesar 93,3% dan 0% untuk nilai *false negative*. Pada tabel tersebut juga dapat diketahui bahwa dalam skenario percobaan yang memiliki 15 dari 15 keberhasilan berarti menandakan bahwa aplikasi berhasil mendeteksi kejadian jatuh secara keseluruhan yang membuat nilai *true positive* adalah 100%. Sebaliknya pada skenario dengan nilai *true positive* yang hanya 93,3% berarti aplikasi sempat tidak mendeteksi kejadian jatuh sekali dari 15 total percobaan.

Sesuai dengan metode *confusion matrix*, maka untuk menemukan rata-rata nilai *true positive* dan *false negative* adalah sebagai berikut.

5.2.2.2. Skenario Tidak Jatuh

Seperti telah dijelaskan, beberapa skenario tidak jatuh diujikan untuk melihat bagaimana akurasi aplikasi dalam menangkap gerakan tidak jatuh. Selain itu, skenario terdiri dari aktivitas-aktivitas pasif maupun aktif meliputi duduk, lompat, berlari, berjalan, bangun dari tidur, hingga aktivitas kombinasi dari aktivitas-aktivitas dasar tersebut. Pengujian ini bertujuan untuk mengetahui apakah algoritma pendeteksian jatuh memiliki kesalahan deteksi pada gerakan yang bukan jatuh.

Tabel 5.9 merupakan detail hasil uji coba yang diperoleh dari pengujian dengan skenario masing-masing. Untuk data yang dihasilkan dari pengujian ini, dapat dilihat pada seperti berikut:

Tabel 5.9 Pengujian Akurasi Aplikasi Dengan Skenario Tidak Jatuh

Skenario Tidak Jatuh	Jumlah Percobaan	True Negative (TN)	False Positive (FP)	% TN	% FP
Duduk	15	15	0	100	0
Melompat	15	15	0	100	0

Skenario Tidak Jatuh	Jumlah Percobaan	True Negative (TN)	False Positive (FP)	% TP	% TN
Berlari	15	15	0	100	0
Bangun Tidur	15	15	0	100	0
Kombinasi	10	9	1	90	10
berdiri/ berjalan menjadi duduk	15	14	1	93,3	0,66
berdiri/ berjalan/ duduk menjadi posisi tidur (sejajar lantai)	15	13	2	86,6	13,3
Total	**100**	**96**	**4**	**669,0**	**10**

5.2.2.3. Confusion Matrix Deteksi Jatuh

Seperti telah dijelaskan pada subbab sebelumnya, perhitungan akurasi kali ini menggunakan *confusion matrix* Dari Tabel 5.8 dan Tabel 5.9 telah didapatkan semua nilai *true positive*, *false positive, true negative*, dan *false negative*. Setelah nilai-nilai tersebut berhasil didapat, dapat dibuat sebuah tabel yang merupakan *confusion of table* dari hasil uji coba seperti dapat dilihat pada Tabel 5.10

Tabel 5.10 Matriks Confusion Hasil Uji Coba Algoritma

		Hasil Deteksi	
		Jatuh	Tidak Jatuh
Kejadian Sebenarnya	Jatuh	**58**	**2**
	Tidak Jatuh	**4**	**96**

Dari matriks *confusion* di atas, dapat diketahui dari 60 total kejadian jatuh, aplikasi mampu mengklasifikasikan 58 kejadian sebagai kejadian jatuh, sedangkan dua kejadian jatuh terklasifikasi sebagai tidak jatuh. Di sisi lain, dari total 70 kejadian tidak jatuh,

aplikasi salah mengklasifikasikan satu kejadian sebagai kejadian jatuh dan 69 lainnya benar terdeteksi sebagai tidak jatuh.

Dari hasil *confusion matrix* yang telah didapat, bisa didapatkan nilai sensivitas , spesifitas, PPV, dan NPV sebagai berikut:

- Nilai **Sensivitas** $= \frac{\text{TP}}{\text{TP+FN}} \times 100\%$
  $= \frac{58}{58+2} \times 100\%$
  $= \frac{58}{60} \times 100\%$
  $= 96{,}6\%$
- Nilai **Spesifitas** $= \frac{\text{TN}}{\text{FP+TN}} \times 100\%$
  $= \frac{96}{4+96} \times 100\%$
  $= \frac{96}{100} \times 100\%$
  $= 96{,}6\%$
- Nilai **PPV** $= \frac{\text{TP}}{\text{TP+FP}} \times 100\%$
  $= \frac{58}{58+4} \times 100\%$
  $= \frac{58}{62} \times 100\%$
  $= 93{,}5\%$
- Nilai **NPV** $= \frac{\text{TN}}{\text{FN+TN}} \times 100\%$
  $= \frac{96}{2+96} \times 100\%$
  $= \frac{96}{98} \times 100\%$
  $= 97{,}9\%$

Kemudian, untuk menentukan tingkat akurasi algoritma pendeteksi jatuh yang diimplementasikan pada aplikasi, adalah dengan membagi semua nilai deteksi yang benar dengan semua jumlah percobaan yang dilakukan. Untuk lebih jelasnya akan dijelaskan pada penjelasan berikut:

- Nilai **Akurasi** $= \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \times 100\%$
  $= \frac{58+96}{58+4+96+2} \times 100\%$

$$= \frac{127}{130} \times 100\%$$
$$= 96{,}3\%$$

5.3. Evaluasi Pengujian

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian kebutuhan fungsionalitas dan evaluasi pengujian akurasi deteksi jatuh.

5.3.1. Evaluasi Pengujian Fungsionalitas

Ringkasan mengenai hasil pengujian fungsionalitas dapat dilihat pada Tabel 5.10. Berdasar data yang ditunjukkan pada tabel tersebut dapat diketahui bahwa semua pengujian fungsionalitas aplikasi berhasil secara keseluruhan.

Tabel 5.11 Ringkasan Evaluasi Hasil Pengujian Fungsionalitas

ID Pengujian	Nama	Hasil
SP-Proses-1.1	Mendaftar Pada Sistem	Berhasil
SP-Proses-1.2	Masuk Dalam Sistem	Berhasil
SP-Proses-1.3	Mengaktifkan Kinect	Berhasil
SP-Proses-1.4	Mendeteksi Jatuh	Berhasil
SP-Proses-1.5	Mengelola Data Lansia	Berhasil

5.3.2. Evaluasi Pengujian Akurasi

Dari subbab skenario dan hasil pengujian sebelumnya, setelah mendapatkan *confusion matrix* dari hasil uji coba telah didapatkan nilai-nilai sensitivitas, spesifitas, *positive* predictive, *negative predictive*, dan tingkat akuasi dari algoritma *decision tree* deteksi jatuh. Untuk lebih detailnya, evaluasi pengujian akurasi dapat dilihat pada Tabel 5.12 berikut:

Tabel 5.12 Evaluasi Pengujian Akurasi Deteksi Jatuh

Nama	Nilai
Nilai Sensivitas	96,6%
Nilai Spesifitas	96,6%

Nama	Nilai
Nilai *Positive Predictive*	93,5%
Nilai *Negative Predictive*	97,9%
Tingkat Akurasi	96,3%

Dari tabel di atas telah didapat beberapa informasi penting, diantaranya adalah nilai sensivitas dan spesifitas algoritma, yang memiliki nilai sebanyak 96,6% dan 98,6%. Selain itu, algoritma yang di uji juga memiliki nilai presisi atau *positive predictive* dan *negative predictive* yakni sebanyak 98,3% dan 97,1%. Sedangkan untuk tingkat akurasi algoritma ditunjukkan pada nilai akurasi hasil uji coba yang telah dilakukan dan didapat nilai 97,7%.

LAMPIRAN

Pada bab ini akan diberikan lampiran yang memberikan informasi tambahan hasil uji coba akurasi deteksi. Sebelum mlihat tabel lampiran, perlu diperhatikan beberapa pemisalan variabel. Diantaranya adalah sebagai berikut:

Ai = (dAi-1)	**Aj** = (dAi)	**Ak** = (deltadA)	**Al** = (vAi)
Bi = (dBi-1)	**Bj** = (dBi)	**Bk** = (deltadB)	**Bl** = (vBi)
Ci = (dCi-1)	**Cj** = (dCi)	**Ck** = (deltadC)	**Cl** = (vCi)
Di = (dDi-1)	**Dj** = (dDi)	**Dk** = (deltadD)	**Dl** = (vDi)
Vf = (VFall)		**X =** Label Kelas	

Tabel 8.1 Lampiran Tabel Hasil Uji Coba Skenario Jatuh (a)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,071	2,042	-0,03	-0,85	-0,42	1,989	1,961	-0,03	-0,85	-0,46	1,985	1,95	-0,03	-1,05	-0,43	2,013	1,991	-0,02	-0,65	-0,42	0,431	1
2,058	2,026	-0,03	-0,98	-0,54	2,001	1,988	-0,01	-0,41	-0,41	1,989	1,931	-0,06	-1,79	-0,52	2,001	1,972	-0,03	-0,87	-0,5	0,493	1
2,01	1,983	-0,03	-0,83	-0,54	2,018	1,995	-0,02	-0,71	-0,42	1,918	1,891	-0,03	-0,8	-0,57	1,98	1,955	-0,03	-0,77	-0,5	0,509	1
2,087	2,044	-0,04	-1,3	-0,6	2,031	2,046	0,014	0,427	-0,6	2,005	1,968	-0,04	-1,15	-0,61	2,042	2,008	-0,03	-1,04	-0,58	0,598	1
2,073	2,044	-0,03	-0,87	-0,7	1,998	2,016	0,018	0,545	-0,69	1,98	1,955	-0,03	-0,76	-0,66	2,02	1,993	-0,03	-0,8	-0,67	0,68	1

Tabel 8.2 Lampiran Tabel Hasil Uji Coba Skenario Jatuh (b)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,065	2,026	-0,04	-1,17	-0,38	2,055	2,026	-0,03	-0,86	-0,48	1,985	1,96	-0,02	-0,73	-0,42	2,023	1,991	-0,03	-0,93	-0,39	0,417	1
2,054	2,025	-0,03	-0,86	-0,61	2,005	2,066	0,061	0	-0,59	1,99	1,98	-0,01	-0,3	-0,48	2,037	2,016	-0,02	-0,66	-0,51	0,544	1
2,068	2,026	-0,04	-1,28	-0,45	2,03	2,006	-0,02	-0,73	-0,41	1,986	1,942	-0,04	-1,33	-0,45	2,014	1,985	-0,03	-0,88	-0,45	0,441	1
1,97	1,968	-0	-0,07	-0,9	1,911	1,912	0,001	0,037	-0,92	2,007	2,006	-0	-0,02	-0,67	1,968	1,97	0,002	0,051	-0,78	0,818	1
1,989	1,957	-0,03	-0,97	-0,82	1,951	1,924	-0,03	-0,81	-0,75	1,974	1,948	-0,03	-0,81	-0,54	1,976	1,952	-0,02	-0,75	-0,72	0,709	1
1,984	1,986	0,002	0,046	-0,81	2,043	2,041	-0	-0,06	-0,57	1,961	1,965	0,004	0,126	-0,64	1,983	1,981	-0	-0,07	-0,68	0,677	1
2,036	2,016	-0,02	-0,61	-0,7	1,943	1,926	-0,02	-0,53	-0,63	2,028	2,015	-0,01	-0,38	-0,67	2,002	1,983	-0,02	-0,57	-0,65	0,664	1
2,055	2,039	-0,02	-0,46	-0,72	2,008	1,993	-0,01	-0,43	-0,55	1,972	1,919	-0,05	-1,57	-0,66	2,003	1,985	-0,02	-0,53	-0,66	0,649	1
1,776	1,774	-0	-0,05	-0,84	1,791	1,789	-0	-0,05	-0,63	1,846	1,846	0	-0	-0,49	1,83	1,828	-0	-0,05	-0,57	0,633	1
2,087	2,08	-0,01	-0,22	-0,74	2,089	2,081	-0,01	-0,25	-0,54	2,088	2,08	-0,01	-0,25	-0,48	2,051	2,041	-0,01	-0,29	-0,71	0,618	1
2,287	2,289	0,002	0,052	0,035	2,21	2,212	0,002	0,064	0,069	2,176	2,18	0,003	0,1	0,075	2,217	2,218	0,002	0,055	0,038	0,054	0
2,154	2,148	-0,01	-0,18	-0,6	2,077	2,081	0,003	0,097	-0,61	2,073	2,064	-0,01	-0,27	-0,61	2,11	2,106	-0	-0,13	-0,58	0,601	1
1,979	1,953	-0,03	0,027	-0,69	1,906	1,906	7E-04	-0	-0,56	1,987	1,981	-0,01	0,006	-0,42	1,942	1,94	-0	0,002	-0,63	0,576	1
2,053	2,03	-0,02	0,023	-0,61	2,055	2,047	-0,01	0,008	-0,31	1,981	1,956	-0,03	0,026	-0,8	2,016	1,999	-0,02	0,017	-0,53	0,561	1

Tabel 8.3 Lampiran Tabel Hasil Uji Coba Skenario Jatuh (c)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
1,928	1,928	-0	-0,01	-0,6	1,895	1,9	0,005	0,152	-0,66	2,017	2,001	-0,02	-0,49	-0,39	1,938	1,938	4E-04	0,013	-0,45	0,523	1
2,087	2,057	-0,03	-0,87	-0,62	2	1,982	-0,02	-0,54	-0,49	2,115	2,098	-0,02	-0,48	-0,44	2,073	2,046	-0,03	-0,8	-0,52	0,518	1
2,107	2,112	0,005	0,149	-0,51	2,052	2,03	-0,02	-0,66	-0,56	2,084	2,087	0,003	0,086	-0,45	2,054	2,074	0,02	0,603	-0,54	0,515	1
2,077	2,055	-0,02	-0,66	-0,58	2,033	2,008	-0,03	-0,77	-0,42	1,976	1,972	-0	-0,11	-0,52	2,02	2,003	-0,02	-0,52	-0,52	0,509	1
2,093	2,09	-0	-0,1	-0,52	2,066	2,05	-0,02	-0,48	-0,39	2,035	2,023	-0,01	-0,35	-0,57	2,048	2,046	-0	-0,04	-0,53	0,503	1
1,965	1,955	-0,01	-0,3	-0,64	1,899	1,894	-0,01	-0,17	-0,54	1,992	1,975	-0,02	-0,51	-0,36	1,944	1,927	-0,02	-0,5	-0,47	0,503	1
2,108	2,055	-0,05	-1,6	-0,51	2,003	1,968	-0,04	-1,08	-0,47	2,075	2,036	-0,04	-1,17	-0,44	2,059	2,019	-0,04	-1,22	-0,49	0,478	1
2,07	2,053	-0,02	-0,52	-0,51	2,067	2,055	-0,01	-0,35	-0,48	1,996	1,981	-0,01	-0,44	-0,41	2,025	2,016	-0,01	-0,28	-0,48	0,467	1
2,082	2,079	-0	-0,08	-0,51	2,016	2,009	-0,01	-0,23	-0,46	2,05	2,028	-0,02	-0,67	-0,36	2,03	2,022	-0,01	-0,25	-0,48	0,453	1
1,877	1,879	0,002	0,06	-0,55	1,882	1,879	-0	-0,1	-0,35	1,992	1,99	-0	-0,06	-0,42	1,906	1,908	0,002	0,064	-0,46	0,445	1
2,118	2,123	0,005	0,152	-0,5	2,087	2,076	-0,01	-0,3	-0,42	2,096	2,092	-0	-0,1	-0,37	2,083	2,072	-0,01	-0,31	-0,41	0,422	1
1,997	1,998	0,001	0,036	-0,5	1,946	1,946	0	7E-04	-0,41	1,937	1,937	0	-0	-0,34	1,942	1,939	-0	-0,09	-0,41	0,415	1
2,05	2,039	-0,01	0,012	-0,44	1,961	1,957	-0	0,005	-0,4	2,004	2,009	0,006	-0,01	-0,39	1,992	1,997	0,005	-0,01	-0,39	0,405	1
2,202	2,185	-0,02	-0,5	-0,39	2,124	2,108	-0,02	-0,48	-0,45	2,132	2,117	-0,02	-0,47	-0,36	2,142	2,124	-0,02	-0,55	-0,39	0,396	1
2,159	2,172	0,012	0,378	0,237	2,109	2,122	0,012	0,375	0,229	2,069	2,077	0,008	0,234	0,228	2,085	2,095	0,01	0,303	0,231	0,231	0

Tabel 8.4 Lampiran Tabel Hasil Uji Coba Skenario Jatuh (d)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
1,956	1,956	0	-0	-0,59	1,946	1,948	0,002	0,064	-0,51	1,998	2,002	0,005	0,136	-0,04	1,974	1,974	-0	-0,02	-0,4	0,384	1
2,181	2,16	-0,02	-0,65	-0,33	2,126	2,106	-0,02	-0,6	-0,35	2,101	2,098	-0	-0,11	-0,5	2,145	2,128	-0,02	-0,5	-0,34	0,38	1
1,992	1,965	-0,03	-0,83	-0,38	1,914	1,899	-0,01	-0,43	-0,51	2,016	1,992	-0,02	-0,72	-0,33	1,967	1,944	-0,02	-0,68	-0,29	0,378	1
2,102	2,091	-0,01	-0,34	-0,39	2,059	2,051	-0,01	-0,25	-0,43	2,036	2,026	-0,01	-0,31	-0,29	2,04	2,028	-0,01	-0,35	-0,38	0,373	1
2,127	2,118	-0,01	-0,25	-0,4	2,101	2,087	-0,01	-0,44	-0,42	2,12	2,096	-0,02	-0,74	-0,24	2,097	2,083	-0,01	-0,41	-0,4	0,364	1
2,047	2,018	-0,03	-0,89	-0,36	1,961	1,952	-0,01	-0,3	-0,37	1,946	1,943	-0	-0,1	-0,37	1,976	1,962	-0,01	-0,43	-0,32	0,355	1
2,023	2,026	0,003	0,089	-0,27	1,929	1,93	0,001	0,039	-0,56	2,016	2,013	-0	-0,1	-0,24	1,979	1,979	-0	-0,01	-0,35	0,355	1
1,994	1,995	0,001	0,032	-0,46	2,04	2,04	-0	-0,02	-0,3	1,965	1,965	0	7E-04	-0,27	1,983	1,983	-0	-0	-0,38	0,352	1
2,072	2,036	-0,04	-1,04	-0,39	1,972	1,943	-0,03	-0,84	-0,36	2,041	2,028	-0,01	-0,38	-0,23	2,023	2,002	-0,02	-0,61	-0,38	0,339	1
2,024	2,011	-0,01	-0,39	-0,43	1,962	1,963	8E-04	0,024	-0,4	2,038	2,037	-0	-0,03	-0,11	1,982	1,988	0,006	0,169	-0,41	0,337	1
1,987	1,987	6E-04	0,019	-0,28	1,917	1,931	0,014	0,436	-0,48	1,994	1,977	-0,02	-0,54	-0,25	1,959	1,943	-0,02	-0,47	-0,3	0,326	1
2,199	2,185	-0,01	-0,44	-0,35	2,161	2,154	-0,01	-0,23	-0,37	2,155	2,109	-0,05	-1,37	-0,21	2,132	2,123	-0,01	-0,26	-0,34	0,32	1
1,984	1,972	-0,01	-0,35	-0,29	1,899	1,898	-0	-0,03	-0,16	1,987	1,952	-0,03	-1,05	-0,47	1,936	1,924	-0,01	-0,35	-0,34	0,315	1
2,127	2,087	-0,04	-1,21	-0,32	2,118	2,089	-0,03	-0,88	-0,36	2,095	2,088	-0,01	-0,19	-0,28	2,092	2,051	-0,04	-1,26	-0,29	0,313	1
1,955	1,954	-0	-0,04	-0,31	1,907	1,908	2E-04	0,007	-0,3	1,982	1,982	0	0,001	-0,3	1,933	1,933	4E-04	0,013	-0,32	0,309	1

Tabel 8.5 Lampiran Tabel Hasil Uji Coba Skenario Jatuh (e)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,011	1,999	-0,01	-0,36	-0,39	1,963	1,963	6E-04	0,016	-0,29	2,037	2,038	8E-04	0,023	-0,29	1,988	1,982	-0,01	-0,15	-0,29	0,312	1
2,025	2,024	-0	-0,02	-0,24	1,917	1,921	0,003	0,098	-0,42	2	1,993	-0,01	-0,23	-0,27	1,977	1,981	0,003	0,098	-0,33	0,315	1
2,057	2,042	-0,01	-0,45	-0,42	2,013	2,005	-0,01	-0,25	-0,45	1,984	1,963	-0,02	-0,64	-0,39	1,989	1,976	-0,01	-0,38	-0,42	0,419	1
2,06	2,045	-0,02	-0,46	-0,47	1,987	1,972	-0,02	-0,45	-0,46	1,987	1,97	-0,02	-0,51	-0,39	1,989	1,974	-0,02	-0,46	-0,48	0,451	1
1,942	1,944	0,001	0,04	0,591	1,899	1,902	0,003	0,09	0,05	1,849	1,868	0,019	0,552	0,335	1,896	1,895	-0	-0,03	0,259	0,309	1
2,058	2,05	-0,01	-0,25	-0,44	1,965	1,962	-0	-0,1	-0,45	1,968	1,96	-0,01	-0,25	-0,43	1,988	1,979	-0,01	-0,26	-0,44	0,438	1
2,057	2,04	-0,02	-0,5	-0,56	2,036	2,019	-0,02	-0,47	-0,56	1,998	2,009	0,011	0,315	-0,58	2,029	2,009	-0,02	-0,61	-0,48	0,543	1
2,056	2,045	-0,01	-0,34	-0,37	1,993	1,962	-0,03	-0,93	-0,38	1,961	1,954	-0,01	-0,19	-0,35	1,983	1,975	-0,01	-0,25	-0,37	0,368	1
2,054	2,041	-0,01	-0,4	-0,45	1,97	1,964	-0,01	-0,19	-0,47	1,966	1,956	-0,01	-0,31	-0,45	1,988	1,974	-0,01	-0,41	-0,48	0,462	1
2,065	2,041	-0,02	-0,7	-0,55	2,014	1,999	-0,02	-0,44	-0,57	1,977	1,955	-0,02	-0,64	-0,52	1,992	1,971	-0,02	-0,62	-0,55	0,547	1
2,065	2,049	-0,02	-0,46	-0,45	2,028	1,995	-0,03	-0,94	-0,4	1,983	1,968	-0,01	-0,41	-0,45	2,01	1,982	-0,03	-0,8	-0,42	0,429	1

Tabel 8.6 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (a)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,359	2,361	0,002	0,059	0,11	2,308	2,309	0,001	0,035	0,081	2,277	2,278	9E-04	0,027	0,117	2,295	2,297	0,002	0,057	0,121	0,107	0
2,355	2,352	-0	-0,08	-0,15	2,335	2,331	-0	-0,13	-0,07	2,307	2,303	-0	-0,12	-0,1	2,324	2,316	-0,01	-0,24	-0,11	0,107	0
1,913	1,914	6E-04	0,016	-0,19	1,923	1,923	7E-04	0,02	0,162	2,002	2,002	-0	-0	0,015	1,949	1,952	0,003	0,086	-0,05	0,105	0
2,357	2,36	0,003	0,099	0,09	2,31	2,314	0,005	0,138	0,132	2,261	2,265	0,003	0,095	0,107	2,286	2,291	0,005	0,143	0,1	0,107	0
2,38	2,377	-0	-0,08	-0,09	2,316	2,312	-0	-0,12	-0,12	2,281	2,279	-0	-0,04	-0,1	2,314	2,311	-0	-0,07	-0,1	0,101	0
2,323	2,324	9E-04	0,025	0,055	2,27	2,271	2E-04	0,005	0,174	2,231	2,232	0,001	0,032	0,086	2,256	2,258	0,002	0,058	0,069	0,096	0
2,287	2,292	0,005	0,159	0,129	2,251	2,254	0,003	0,088	0,187	2,194	2,198	0,004	0,116	0,055	2,223	2,229	0,007	0,193	0,086	0,114	0
2,246	2,246	#######	0,002	-0,06	2,206	2,206	5E-04	0,014	-0,07	2,139	2,139	-0	-0,02	-0,07	2,171	2,171	3E-04	0,008	-0,07	0,067	0
2,393	2,401	0,007	0,212	0,069	2,273	2,276	0,002	0,073	-0,06	2,281	2,283	0,001	0,044	0,079	2,309	2,314	0,004	0,127	0,068	0,07	0
2,379	2,38	0,001	0,039	0,076	2,264	2,265	0,001	0,035	-0,13	2,295	2,296	0,001	0,034	0,092	2,308	2,309	9E-04	0,027	0,08	0,094	0
2,335	2,333	-0	-0,06	-0,09	2,322	2,321	-0	-0,02	-0,06	2,285	2,283	-0	-0,07	-0,09	2,314	2,298	-0,02	-0,49	-0,04	0,07	0
2,271	2,273	0,002	0,063	0,092	2,197	2,198	8E-04	0,024	0,052	2,168	2,17	0,002	0,06	0,079	2,197	2,2	0,003	0,085	0,102	0,081	0
2,305	2,307	0,003	0,081	-0,05	2,247	2,25	0,003	0,083	-0,09	2,204	2,207	0,002	0,072	-0,04	2,236	2,239	0,003	0,076	-0,06	0,059	0
1,988	1,986	-0	-0,06	-0,09	1,946	1,945	#######	-0	-0,02	1,935	1,935	-0	-0	-0,03	1,944	1,944	2E-04	0,007	-0,1	0,058	0

Tabel 8.7 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (b)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,285	2,284	-0	-0,02	0	2,222	2,222	#### #	-0	0	2,187	2,186	-0	-0,03	0	2,212	2,21	-0	-0,07	0	0	0
-0,01	-0,01	-0	-0,1	-0,04	-0,03	-0,03	-0	-0,02	-0,07	-0,02	-0,02	0,004	0,122	-0,01	-0,02	-0,02	-0	-0,02	-0,03	0,039	0
2,306	2,301	-0	-0,15	-0,07	2,267	2,262	-0,01	-0,17	-0,08	2,21	2,204	-0,01	-0,17	-0,09	2,24	2,239	-0	-0,05	-0,1	0,084	0
2,289	2,284	-0	-0,15	0,008	2,238	2,234	-0	-0,12	-0,03	2,244	2,242	-0	-0,05	-0,01	2,252	2,249	-0	-0,11	0,015	0,016	0
2,4	2,399	-0	-0,03	-0,03	2,357	2,359	0,002	0,064	0,025	2,352	2,351	-0	-0,02	0,003	2,365	2,365	-0	-0,01	0,006	0,015	0
2,31	2,31	4E-04	-0	0,005	2,25	2,249	-0	9E-04	0,036	2,226	2,226	8E-04	-0	0,018	2,245	2,245	5E-04	-0	0,013	0,018	0
2,361	2,362	7E-04	0,022	0,02	2,309	2,31	1E-03	0,03	0,016	2,255	2,256	8E-04	0,023	0,02	2,288	2,289	5E-04	0,016	0,016	0,018	0
1,997	1,995	-0	-0,06	-0,02	2,031	2,031	-0	-0,03	-0,02	1,964	1,963	-0	-0,03	-0	1,985	1,985	-0	-0,01	-0,02	0,014	0
2,329	2,33	0,001	0,037	-0,01	2,271	2,272	9E-04	0,026	-0,02	2,228	2,23	0,002	0,045	-0,02	2,264	2,266	0,002	0,049	-0,01	0,014	0
2,339	2,338	-0	-0,03	-0,02	2,311	2,311	-0	-0,01	-0,03	2,261	2,26	-0	-0,01	0,002	2,29	2,29	-0	-0,02	0,006	0,014	0
2,329	2,325	-0	-0,12	-0,1	2,255	2,251	-0	-0,12	-0,11	2,241	2,238	-0	-0,09	-0,09	2,265	2,261	-0	-0,11	-0,1	0,097	0
2,21	2,207	-0	-0,1	-0,13	2,175	2,175	-0	-0,02	-0,03	2,165	2,162	-0	-0,08	-0,12	2,165	2,163	-0	-0,08	-0,1	0,096	0
2,489	2,49	7E-04	0,022	0,114	2,412	2,414	0,002	0,05	0,113	2,359	2,358	-0	-0,03	0,109	2,423	2,424	0,001	0,04	0,124	0,115	0

Tabel 8.8 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (c)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,263	2,269	0,006	0,178	-0,07	2,247	2,25	0,004	0,114	-0,07	2,211	2,213	0,002	0,064	-0,05	2,222	2,23	0,008	0,236	-0,05	0,062	0
2,336	2,335	-0	-0,04	-0,06	2,286	2,283	-0	-0,08	-0,06	2,242	2,24	-0	-0,06	-0,05	2,276	2,275	-0	-0,05	-0,06	0,056	0
2,02	2,015	-0,01	-0,17	-0,03	1,993	1,989	-0	-0,13	-0,12	1,959	1,954	-0,01	-0,17	-0,01	1,966	1,963	-0	-0,1	-0,05	0,054	0
2,32	2,317	-0	-0,08	0,055	2,242	2,241	-0	-0,01	0,058	2,223	2,223	7E-04	0,02	0,056	2,251	2,251	-0	-0,02	0,053	0,056	0
1,889	1,89	0,001	0,032	0,006	1,879	1,881	0,003	0,079	-0,01	1,946	1,949	0,003	0,1	-0,14	1,914	1,916	0,002	0,047	0,045	0,052	0
2,288	2,29	0,002	0,069	0,052	2,236	2,225	-0,01	-0,31	0,058	2,181	2,184	0,003	0,103	0,035	2,213	2,215	0,002	0,074	0,057	0,051	0
2,245	2,246	0,002	0,047	0,06	2,186	2,187	0,002	0,049	0,064	2,134	2,135	0,001	0,04	0,068	2,165	2,167	0,002	0,047	0,066	0,064	0
2,418	2,413	-0,01	-0,15	-0,06	2,371	2,369	-0	-0,05	-0,04	2,362	2,358	-0	-0,13	-0,09	2,377	2,374	-0	-0,08	-0,05	0,064	0
2,305	2,307	0,003	0,081	-0,05	2,247	2,25	0,003	0,083	-0,09	2,204	2,207	0,002	0,072	-0,04	2,236	2,239	0,003	0,076	-0,06	0,059	0
2,303	2,304	6E-04	0,019	-0,01	2,247	2,248	0,001	0,031	-0,02	2,236	2,238	0,001	0,042	-0,01	2,238	2,239	7E-04	0,021	-0,01	0,011	0
2,382	2,382	1E-04	0,004	0,01	2,311	2,311	2E-04	0,005	0,003	2,315	2,315	1E-04	0,003	0,006	2,317	2,318	2E-04	0,007	0,008	0,007	0
2,327	2,335	0,008	0,243	0,605	2,286	2,293	0,007	0,207	0,448	2,258	2,264	0,006	0,186	0,156	2,279	2,286	0,007	0,207	0,18	0,347	1
2,274	2,275	0,001	0,037	0,032	2,221	2,222	0,002	0,051	0,027	2,17	2,173	0,002	0,072	0,04	2,198	2,2	0,002	0,056	0,027	0,032	0
2,351	2,355	0,003	0,102	0,031	2,302	2,304	0,002	0,059	0,044	2,272	2,276	0,004	0,111	0,016	2,303	2,306	0,002	0,068	0,035	0,032	0

Tabel 8.9 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (d)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
1,976	1,974	-0	-0,05	-0,02	2,027	2,027	-0	-0,02	0,01	1,944	1,943	-0	-0,03	0,087	1,99	1,989	-0	-0,02	0,008	0,031	0
2,343	2,343	-0	-0,02	-0,03	2,318	2,316	-0	-0,06	-0,01	2,26	2,26	-0	-0,01	-0,02	2,289	2,289	1E-04	0,004	0,003	0,015	0
2,343	2,342	-0	-0,02	0,028	2,274	2,275	5E-04	0,016	0,019	2,255	2,254	-0	-0,05	0,039	2,274	2,273	-0	-0,01	0,024	0,028	0
2,334	2,335	0,001	0,034	0,059	2,268	2,268	9E-04	0,027	0,039	2,235	2,237	0,002	0,052	0,054	2,264	2,265	0,001	0,036	0,056	0,052	0
2,357	2,359	0,002	0,071	0,092	2,236	2,237	5E-04	0,014	0,047	2,24	2,242	0,002	0,059	-0,08	2,274	2,277	0,003	0,096	0,084	0,076	0
2,346	2,347	0,001	0,035	-0	2,294	2,295	1E-03	0,029	-0,03	2,24	2,241	6E-04	0,018	0,001	2,273	2,274	0,001	0,033	-0,01	0,009	0
2,338	2,341	0,004	0,109	0,141	2,294	2,297	0,003	0,076	0,119	2,255	2,258	0,003	0,082	0,077	2,274	2,277	0,003	0,103	0,1	0,11	0
2,345	2,346	0,001	0,038	-0	2,294	2,294	2E-04	0,005	-0,03	2,239	2,24	6E-04	0,018	0,001	2,272	2,273	0,001	0,033	-0,01	0,009	0
2,334	2,334	-0	-0,02	0,039	2,295	2,294	-0	-0,04	0,011	2,24	2,239	-0	-0,03	0,042	2,273	2,272	-0	-0,02	0,044	0,034	0
2,313	2,314	8E-04	0,024	0,001	2,275	2,276	7E-04	0,022	0,002	2,255	2,256	0,001	0,035	-0	2,272	2,273	7E-04	0,022	0,003	0,003	0
2,335	2,336	0,001	0,034	0,032	2,283	2,287	0,004	0,108	0,063	2,236	2,239	0,003	0,098	0,035	2,271	2,273	0,002	0,058	0,034	0,041	0
2,331	2,321	-0,01	-0,31	-0,01	2,268	2,259	-0,01	-0,28	-0,02	2,298	2,292	-0,01	-0,17	0,028	2,271	2,26	-0,01	-0,33	-0,02	0,019	0
2,319	2,318	-0	-0,02	0	2,278	2,277	-0	-0,03	0	2,258	2,258	-0	-0,01	0	2,27	2,268	-0	-0,06	0	0	0

Tabel 8.10 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (e)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,327	2,321	-0,01	-0,17	0,026	2,327	2,325	-0	-0,04	0,088	2,248	2,245	-0	-0,1	0,068	2,269	2,267	-0	-0,06	0,039	0,056	0
2,321	2,316	-0	-0,13	-0,06	2,28	2,278	-0	-0,05	-0,06	2,252	2,25	-0	-0,07	-0,05	2,272	2,269	-0	-0,08	-0,06	0,057	0
2,324	2,33	0,006	0,184	0,2	2,284	2,288	0,003	0,103	0,139	2,249	2,253	0,004	0,131	0,098	2,269	2,274	0,005	0,147	0,146	0,145	0
2,333	2,332	-0	-0,02	0,002	2,281	2,277	-0	-0,13	-0,04	2,236	2,235	-0	-0,02	0,005	2,27	2,269	-0	-0,03	-0	0,012	0
2,272	2,271	-0	-0,03	0,007	2,204	2,202	-0	-0,06	-0,02	2,171	2,17	-0	-0,05	5E-04	2,193	2,193	-0	-0,03	0,005	0,007	0
2,267	2,268	0,002	0,051	0,054	2,214	2,215	0,002	0,045	3E-04	2,164	2,166	0,002	0,046	0,063	2,192	2,193	0,001	0,043	0,058	0,044	0
2,268	2,273	0,005	0,162	0,283	2,217	2,223	0,006	0,174	0,28	2,175	2,183	0,008	0,228	0,259	2,192	2,199	0,007	0,206	0,275	0,274	0
2,257	2,256	-0	-0,03	-0,09	2,175	2,173	-0	-0,06	-0,1	2,173	2,172	-0	-0,02	-0,09	2,195	2,194	-0	-0,04	-0,08	0,091	0
2,265	2,267	0,003	0,075	0,062	2,19	2,19	-0	-0,01	0,047	2,163	2,163	2E-04	0,007	0,07	2,192	2,194	0,003	0,076	0,062	0,06	0
2,267	2,264	-0	0,003	-0,06	2,195	2,193	-0	0,003	-0,09	2,167	2,164	-0	0,003	-0,06	2,186	2,183	-0	0,003	-0,07	0,068	0
2,259	2,255	-0	-0,12	0,031	2,208	2,204	-0	-0,12	0,031	2,165	2,161	-0	-0,12	0,032	2,184	2,18	-0	-0,11	0,028	0,031	0
2,31	2,313	0,003	0,092	0,068	2,235	2,237	0,001	0,044	0,052	2,213	2,214	0,001	0,04	0,061	2,243	2,245	0,002	0,067	0,068	0,062	0
2,345	2,346	0,001	0,036	0,062	2,278	2,278	1E-04	0,004	0,026	2,252	2,255	0,003	0,078	0,064	2,28	2,281	0,002	0,048	0,068	0,055	0

Tabel 8.11 Lampiran Tabel Hasil Uji Coba Skenario Tidak Jatuh (f)

Ai	Aj	Ak	Al	Am	Bi	Bj	Bk	Bl	Bm	Ci	Cj	Ck	Cl	Cm	Di	Dj	Dk	Dl	Dm	Vf	X
2,376	2,376	#### #	-0	0,025	2,316	2,314	-0	-0,07	0,019	2,271	2,272	5E- 04	0,014	0,026	2,306	2,304	-0	-0,06	0,023	0,023	0
2,358	2,358	-0	-0,02	-0,02	2,312	2,311	-0	-0,03	-0,02	2,279	2,279	-0	-0,01	-0,02	2,306	2,305	-0	-0,02	-0,01	0,017	0
2,361	2,367	0,006	0,196	0,205	2,313	2,317	0,004	0,113	0,148	2,283	2,286	0,003	0,097	0,156	2,306	2,312	0,006	0,187	0,18	0,172	0
2,361	2,359	-0	-0,05	-0,03	2,293	2,286	-0,01	-0,23	-0,06	2,287	2,285	-0	-0,05	-0,02	2,305	2,301	-0	-0,12	-0,05	0,041	0

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan tugas akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian perangkat lunak yang dilakukan pada aplikasi pendeteksi jatuh menggunakan Kinect dan *decision tree*, dapat diambil kesimpulan sebagai berikut:

1. Aplikasi mampu menangkap dan mendeteksi tubuh beserta pergerakannya dengan mengimplementasikan teknologi Kinect menggunakan SDK Kinect berupa titik-titik *joint* tubuh.
2. Berdasar hasil uji coba akurasi dapat disimpulkan bahwa Algoritma *decision tree* mampu mendeteksi jatuh dengan akurasi sebesar 97,7%.
3. Fitur-fitur yang digunakan pada *decision tree* antara lain adalah jarak titik *joint* tubuh dengan bidang normal, kecepatan rata-rata titik *joint* tubuh dan rata-rata kecepatan dalam N *frame* yang telah ditentukan.
4. Hasil pengujian deteksi jatuh menunjukkan aplikasi mampu memberikan pemberitahuan terdeteksinya kejadian jatuh pada tangkapan Kinect.
5. Berdasar hasil uji coba fungsionalitas, aplikasi berhasil dibangun sesuai peranangan dan spesifikasi kebutuhan aplikasi.

6.2. Saran

Berikut merupakan saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang sesuai dengan hasil perancangan, implementasi, dan uji coba yang telah dilakukan :

1. Menambah fitur untuk memberikan *alarm* berupa kiriman pesan *short message service* (SMS) maupun *email* pada kerabat atau anggota keluarga lansia ketika terjadi jatuh pada lansia tersebut.
2. Memberikan *alarm* berupa suara. Sehingga ketika pengasuh sedang jauh dan tidak sedang di depan aplikasi bisa segera melakukan pengecekan lebih lanjut.
3. Penambahan *dataset* untuk menghasilkan *decision tree* yang lebih baik dalam mendeteksi dan membedakan antara kejadian jatuh dan tidak jatuh, sehingga angka *false positive* dan *false negative* akan semakin kecil.

DAFTAR PUSTAKA

[1] Center for Disease Control and Prevention (CDC), "Falls among older adults: An overview," 2012. [Online]. Available: http://www.cdc.gov/homeandrecreationalsafety/Falls/adultfalls.html. [Accessed September 2015].

[2] M. Mary E. Tinetti, W. L. Liu and E. B. Claus, "Predictors and prognosis of inability to get up after falls among elderly persons," *J. Amer. Med. Assoc.,* vol. 269, pp. 65-70, 1993.

[3] M. Mary E. Tinetti and M. Christianna S. Williams, "Falls, Injuries Due to Falls, and the Risk of Admission to a Nursing Home," *N Engl J Med,* vol. 337, pp. 1279-1284, 1997.

[4] R. S. Maryam, "Pedoman Pencegahan Jatuh Bagi Lansia di Rumah," *Prodi Keperawatan Persahabatan Jurusan Keperawatan Poltekkes Kemenkes Jakarta,* vol. III, 2013.

[5] Learn Not To Fall, "Fall Facts," 2012. [Online]. Available: http://www.learnnottofall.com/content/fall-facts.jsp. [Accessed September 2015].

[6] Public Health Agency of Canada, "If You Fall or Witness a Fall, Do You Know What to Do?," 2011. [Online]. Available: http://www.phac-aspc.gc.ca/seniors-aines/alt-formats/pdf/publications/public/injury-blessure/falls-chutes/falls-chutes-e.pdf. [Accessed September 2015].

[7] Springer, "Color and Depth Image Correspondence for Kinect v2," in *Advanced Multimedia and Ubiquitous Engineering: Future Information Technology*, New York, Springer, 2015, pp. 111-112.

[8] E. E. Stone and M. Skubic, "Fall Detection in Homes of Older Adults Using the Microsoft Kinect," *IEEE Journal of Biomedical and Health Informatics, Vol. 19, No.1, January, 2015,* vol. 19, 2015.

[9] C. Kawatsu, J. Li and C. Chung, "Development of a Fall Detection System with Microsoft Kinect," 2012.

[10] I. H. Witten, E. Frank and M. A. Hall, Data Mining: Practical machine learning tools and techniques, 3rd Edition, San Francisco: Morgan Kaufmann, 2011.

BIODATA PENULIS

Penulis, **FAHMY THORIQUL HAQ**, lahir di Madiun pada 1 Juli 1996. Ia adalah alumni dari SMA Negeri 2 Kota Madiun. Saat ini, sedang menjalani pendidikan S-1 Teknik Informatika di Institut Teknologi Sepuluh Nopember Surabaya. Selain sibuk dengan kuliah, ia aktif di beberapa organisasi tingkat jurusan sampai institut. Mulai dari himpunan mahasiswa jurusan, badan eksekutif mahasiswa (BEM), hingga organisasi kepemanduan LKMM Fakultas. Sebagai mahasiswa teknologi informasi, ia selalu memiliki ketertarikan pada programming. Beberapa pemrograman yang mulai ditekuni adalah web, mobile application, khususnya game development.Walaupun masih bisa dibilang sebagai pemula yang memasuki dunia pengembangan permainan, penulis akan terus berusaha meningkatkan dan belajar lebih mengenai game development serta mengembangkan dan memajukan wawasan teknologi untuk ikut serta memberi manfaat di sekitar.